Grado en ingeniería electrónica industrial y automática

TRABAJO DE FIN DE GRADO

APLICACIÓN ANDROID PARA LA DETECCIÓN DE LAS LÍNEAS DE LA CARRETERA

Autor:

CARLOS GARCÍA DURÁN

Tutor:

JUAN CARMÓNA FERNÁNDEZ

Septiembre 2015

Si se cree y se trabaja, se puede.

Diego Pablo Simeone, “El Cholo”, Mayo 2014

A mis padres, por sus sacrificios, sin los cuales esto habría sido imposible.

A mi hermano por tantos buenos momentos.

A Sandra por apoyarme y ayudarme a creer en mí mismo.

A mis compañeros que han hecho los momentos duros más llevaderos.

A Fernando por enseñarme a dar los primeros pasos en la materia y a Juan por ayudarme en todo lo que he necesitado.

RESUMEN.

Desarrollo de una aplicación para cualquier dispositivo móvil Android encaminada a la mejora de la seguridad vial utilizando como elemento principal la cámara de dicho dispositivo. La aplicación capta la imagen a través de la cámara y analiza cada ´frame´ de forma que clasificará los tipos de línea de la carretera presentes en cada momento. Distinguirá entre las líneas continuas y discontinuas con el objetivo futuro de que la aplicación nos pueda avisar en caso de que estemos realizando una maniobra de forma incorrecta.

El objetivo es conseguir una aplicación que esté al alcance de todo el mundo ya que lo único que se necesita es un dispositivo móvil Android, tan común en la actualidad. Además de abaratar el coste de esta tecnología, ya que de ir implantada en un automóvil incrementaría su coste notablemente, poderla tener presente en vehículos actuales que no estén dotados de dicha tecnología.

Esta investigación está encaminada a, una vez perfeccionada la tecnología, poderla integrar con otras con el mismo objetivo de incrementar la seguridad vial de forma que se tenga una única aplicación que recoja el mayor número de funciones (detección de líneas, de señales, distancia a otros vehículos, etc.).

Todo esto se ha realizado con la ayuda de las librerías OpenCV, creadas por Intel para el desarrollo de la visión artificial que cuentan con un gran número de funciones las cuales de adecúan a nuestro objetivo. Y desarrollado con el software Eclipse adaptado para Android.

Si se consiguiera desarrollar esta tecnología lo suficiente y tener este tipo de aplicaciones tan necesarias, encaminadas a la seguridad vial, fácilmente al alcance de la mano sería un gran avance para los consumidores tanto desde el punto de vista económico como de su propia seguridad.

ÍNDICE.

ÍNDICE DE IMÁGENES.

ÍNDICE DE TABLAS.

1. INTRODUCCIÓN.

El proyecto desarrollado a continuación parte de una aplicación previamente implementada por el Laboratorio de Sistemas Inteligentes (LSI) de la Universidad Carlos III de Madrid en el campo de la visión por computador aplicada en teléfonos inteligentes desarrollado en el sistema operativo de Android.

El trabajo explicado en la memoria cuenta como se ha realizado, en una nueva subaplicación dentro de la ya existente encaminada a la seguridad vial, la detección y clasificación de las líneas de la carretera en continuas o discontinuas.

Se ha realizado en el entorno de trabajo Android por ser un sistema operativo popular y gratuito y para el tratamiento de la imagen se ha elegido las librerías OpenCV [1], actualmente en auge. Además ha sido implementado en lenguaje Java desde el entorno de desarrollo Eclipse adaptado para Android.

Antes de proceder con el desarrollo de la aplicación se ha hecho un estudio del estado del arte lo más exhaustivo posible en el que se han recogido distintas aplicaciones ya existentes relacionadas con la visión artificial y la seguridad vial.

A continuación, ya nos encontrábamos en disposición de realizar una correcta implementación de la aplicación y de los distintos algoritmos que se aplicarán a cada uno de los frames captados por la cámara de nuestro dispositivo.

2. ESTADO DEL ARTE.

A continuación se recogen los avances del conocimiento en la actualidad que se tratarán y las tendencias existentes.

2.1 ANDROID

Se trata de un sistema operativo pensado en un principio para dispositivos móviles de forma similar a los sistemas iOS, Symbian y Blackberry OS. Está basado en el núcleo del sistema operativo libre, gratuito y multiplataforma de Linux [2].

Este sistema permite crear aplicaciones en Dalvik, una variación de Java. Este sistema operativo proporciona las interfaces necesarias para poder desarrollar aplicaciones que puedan acceder a las distintas funciones del teléfono como el GPS, llamadas, agenda, etc., todo ello de forma muy sencilla con el lenguaje de programación Java. Combina dos características fundamentales para los dispositivos móviles: permite crear aplicaciones con mejor rendimiento y un bajo consumo de energía. Dos características fundamentales debido a que los dispositivos a los que va destinado funcionan con baterías limitadas.

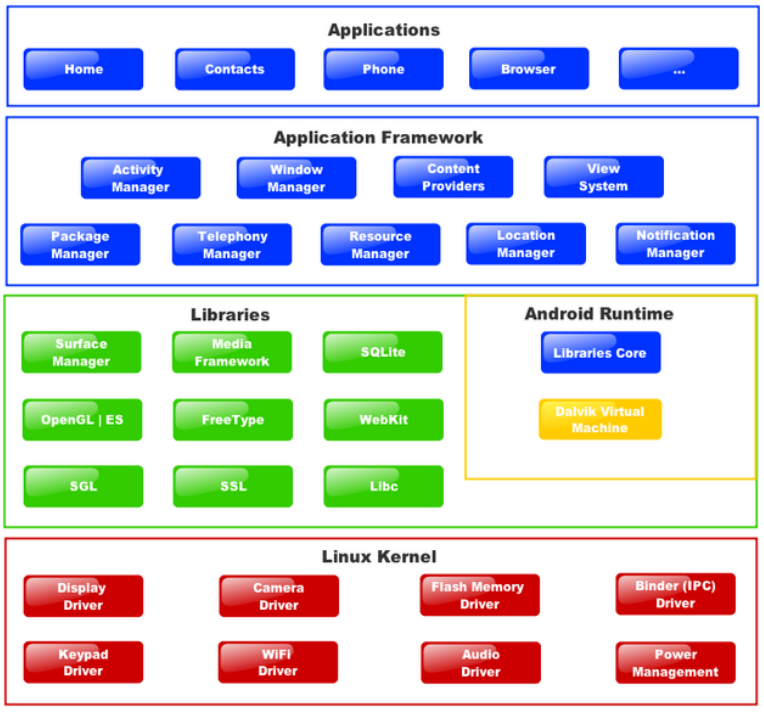


Ilustración 1. Arquitectura de Android

Una de las principales características de Android es que se trata de un sistema operativo completamente gratuito tanto a la hora de programar como a la hora de incluirlo en un teléfono. Por este motivo es por el que los fabricantes y desarrolladores eligen este sistema operativo; los bajos costes tanto de lanzamiento de un teléfono o de una aplicación.

El hecho de que cualquiera pueda bajarse, inspeccionar, compilar y cambiar el código fuente da seguridad a los usuarios, además de, permitir detectar fallos más rápidamente. También permite adaptar mejor el sistema operativo a cada terminal.

Por estos motivos se ha optado por la utilización del sistema operativo Android para este proyecto, tanto por su ventaja a nivel comercial como a nivel técnico (las librerías que permiten ser implementadas y la versatilidad de dicho sistema).

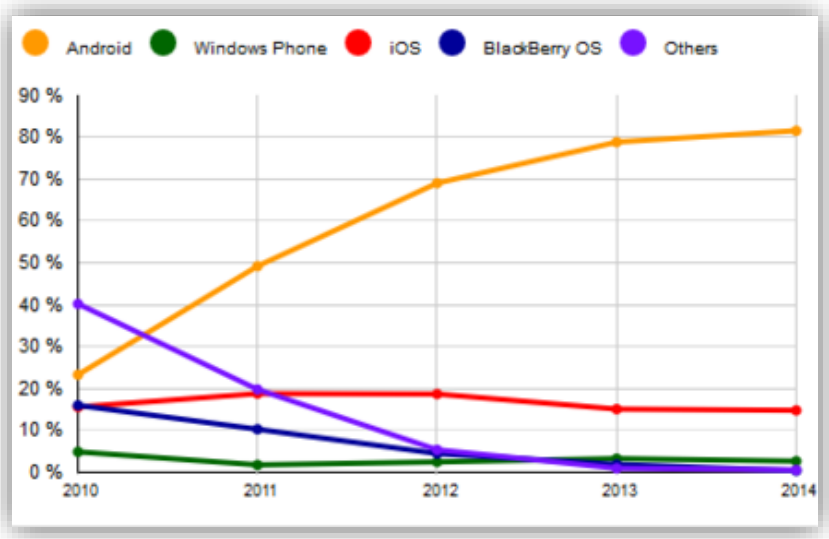


Ilustración 2. Cuota de mercado de los sistemas operativos.

La versión de Android utilizada para este trabajo es la 4.2.1 Jelly Bean.

2.2 Inteligencia artificial

El término de inteligencia artificial (IA) se puede definir como la ciencia que tiene como objetivo imitar el comportamiento inteligente de las personas a través del diseño y construcción de máquinas. Se trata de uno de los retos más fascinantes en el área de las ciencias de la computación [3].

En un principio nació como un estudio filosófico que buscaba imitar la inteligencia humana. Las primeras aplicaciones que aplicaban inteligencia artificial fueron desarrolladas en la época de los 70´s y 80´s dando creación a los sistemas expertos. No obstante, en este periodo surgen numerosas limitaciones en cuanto a este campo de investigación por lo que se llegó a una época de decepción. A partir de los 90´s se conoce como era moderna. Actualmente, se está en una etapa de redefinición donde se da prioridad a la comprensión de la Inteligencia Humana.

La inteligencia artificial es una disciplina que se dedica al desarrollo de software informático capaz de ejecutar trabajos inteligentes. Sus principales objetivos son:

1- Estudiar el comportamiento inteligente de las personas humanas.
2- Crear programas computacionales inteligentes capaces de imitar el comportamiento humando.

En ciertos campos de aplicación, como es el caso de los sistemas computacionales inteligentes y sus algoritmos, los computadores han superado ampliamente la destreza humana. Sin embargo, en otras áreas como la del lenguaje natural, la visión o el aprendizaje por experiencia aún no se ha sido capaz de desarrollar la suficiente habilidad.

La inteligencia artificial funciona mediante algoritmos programados en lenguaje de computadora como pueden ser Lips y Prolog (algunos de los más utilizados) o lenguajes de muy bajo nivel como assembler (lenguaje con el que se comenzó a desarrollar los softwares de IA). Con el desarrollo de estas tecnologías se han ido desarrollando otros lenguajes más adaptables y sencillos.

Para resolver este tipo de problemas relacionados con la inteligencia artificial hace falta tener en cuenta varios puntos antes de comenzar a desarrollar una solución:

1- Tener en cuenta cual es la situación original de la que se parte.
2- Establecer el estado final o solución del problema.
3- Tener claro cuál es el conjunto de operadores y medios que se van a utilizar para que el problema se resuelva.
4- Conocer los posibles espacios o situaciones que se podrán dar en un futuro.
5- Buscar caminos que lleven desde el estado inicial al final.

Estos puntos son fundamentales para un correcto desarrollo del problema.

2.2.1 Ramificaciones de la inteligencia artificial.

Las numerosas aplicaciones que puede llegar a tener la inteligencia artificial hace que sea posible hacer distinción entre sus campos de aplicación:

- **Sistemas expertos**: Se tratan de sistemas de decisión para resolver problemas complejos basados en los conocimientos de un experto. Son programas informáticos que simulan el proceso de pensamiento de una persona experta en determinada área haciendo las veces de intermediario entre dicho experto, que transmite sus conocimientos al sistema, y el usuario del sistema que lo utiliza para resolver cierto problema.

- **Aprendizaje y razonamiento automático**: Se trata de crear sistemas informáticos capaces de modelar los procesos de aprendizaje en sus múltiples manifestaciones utilizando distintos datos o experiencias previas.

- **Robótica**: Robots o autómatas capaces de recibir información a través del exterior, ya sea a través de sensores, cámaras, etc., o de un manipulador humano. A través de esta información deberán ser capaces de modelar el universo que les rodea y en consecuencia determinar un plan para ejecutar las órdenes correctamente. Deben ser capaces de prever las consecuencias de sus futuribles acciones y en caso de resultar inútiles o perjudiciales y actuar en consecuencia.

Ilustración 3. Coche autónomo de Google.

Uno de los principales problemas y retos a los que se enfrenta esta rama de la inteligencia artificial es el de la visión. Mientras que por un lado resulta sencillo interpretar la información obtenida a través de un sensor, no ocurre lo mismo con las imágenes.

Se ha conseguido realizar el reconocimiento de distintas formas predefinidas en el sistema para, por ejemplo, la manipulación de piezas en entornos controlados, sin embargo, resulta mucho más complicado analizar las imágenes de un entorno más abierto y con numerosas posibilidades a las que debe enfrentarse el sistema.

- **Procesamiento de lenguaje natural:** Se trata del análisis de los patrones del lenguaje e intentar crear algoritmos sobre los cuales el computador entienda en palabras utilizadas en una conversación de alto nivel, abstrayendo datos y comprendiendo el mensaje.

- **Redes neuronales:** Se basan en la estructura neuronal del cerebro y su intento de reproducir sus capacidades. Cuenta con un gran número de procesadores que funcionan de forma paralela, cada uno de los cuales se encarga de solucionar una pequeña parte de un problema mayor. Tiene la capacidad de aprendizaje y este será mayor cuanto mayor sea el número de muestras.

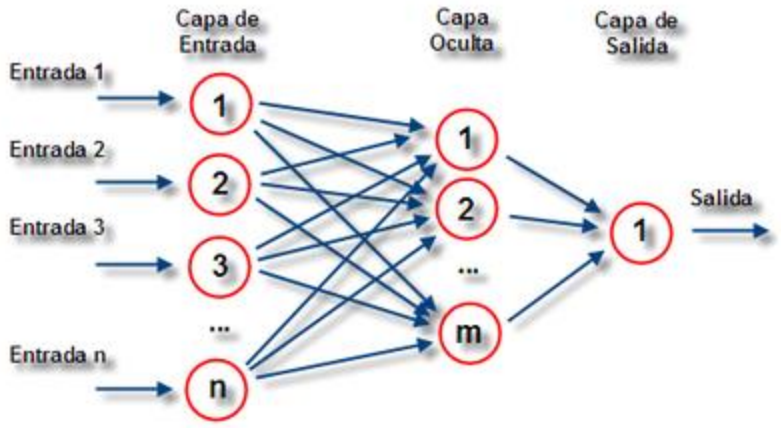


Ilustración 4. Funcionamiento redes neuronales.

- **Algoritmos genéticos:** Proporcionan un método de aprendizaje basado en la analogía con la evolución de las especies. Estos algoritmos generan un conjunto de hipótesis mediante la recombinación de parte del conjunto de hipótesis conocido. En cada paso el conjunto de hipótesis conocido como "población actual" se renueva reemplazando una proporción de esta población por los sucesores de las hipótesis más adecuadas.

Se puede concluir que la inteligencia artificial es un campo que estudia el comportamiento del cerebro humano y lo intenta simular de manera artificial en robots y computadoras.

2.3 Visión por computador

Es una rama de la Inteligencia Artificial dedicado a, a través de computadoras, extraer información de las imágenes y actuando y ofreciendo soluciones a partir de ellas.

Tiene un carácter multidisciplinar y puede aplicarse a numerosos campos como robótica, procesamiento de señales, reconocimiento de patrones, neurociencia, teoría de control y muchos más, desempeñando tareas de reconocimiento, seguimiento de objetos, reconstrucción de escenas, restauración de imágenes o control de calidad entre otros.

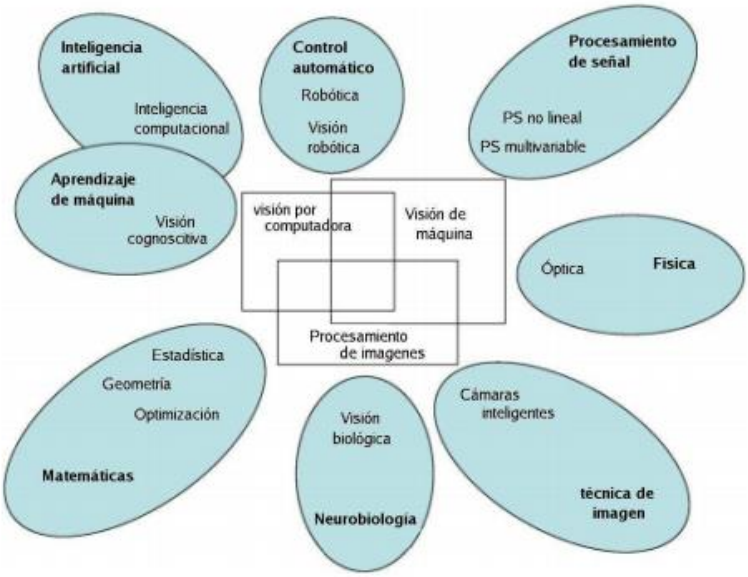


Ilustración 5. Relaciones entre visión por computador y otras áreas afines.

A continuación podemos ver los hechos más relevantes en la visión por computador desde su creación:

Año	Evento
1955	Oliver Selfridge, padre de la percepción por computador describe el primer programa que aprendía y reconocía letras (Reconocimiento de Patrones) [49] [36]
1960	Mejoramiento de las imágenes espaciales a través de procesamiento de imágenes.
1965	Primer Sistema de Visión por computador: *Roberts' Program -- Blocks World* [37], paradigma 2D->3D.
1968	Primer edición de la revista oficial en Reconocimiento de Patrones [38]
1969	Primer Libro en Visión por Computador: *Picture Processing by Computer (A. Rosenfeld)* [39]
1970	Primera iniciativa de la International Conference on Pattern Recognition (ICPR) fundada después 1973. [40]
1977	Primera conferencia IEEE en Computer Vision and Pattern Recognition (CVPR) [41]
1978	Se funda la Asociación Internacional de Reconocimiento de Patrones (IAPR) [42]
1979	Análisis y estimación del movimiento.
1980	Agentes Visuales, primer algoritmo de procesamiento de imagen [43]
1981	Métodos de Optical Flow (L-K y H-S)
1988	Avances en visión 3D. Surge un nuevo campo llamado Visión Activa [44] [45]
1990-2000	Estimadores de movimiento, fondos y posiciones, nuevos filtros en visión artificial (Kalman), face detection, machine learning, realidad aumentada [46], entre otros.
2000+	Visión Cognitiva [47], entre otros

Ilustración 6. Cronología visión por computador.

En la actualidad, en muchas ocasiones, la visión por computador no es la mejor solución para resolver ciertos problemas debido a la complejidad de ciertas imágenes y el exceso de información en ellas. Por ejemplo, imaginemos una carretera con tráfico repleta de

coches. Sin embargo, las soluciones humanas pueden llegar a ser subjetivas e inexactas y en ocasiones lentas.

El sistema de visión humano es capaz de describir fácilmente texturas, bordes, colores, representaciones bidimensionales a partir de una tridimensional y distinguir entre las distintas personas, colores, formas, etc. de una imagen. Pueden vigilar ciertas zonas, distinguir enfermedades a partir de radiografías, etc. Muchas de estas tareas pueden llevarse a cabo con visión artificial bajo el software y el hardware correcto.

A pesar de las limitaciones actuales (complejidad de ciertas imágenes, nivel de inteligencia que puede ofrecer la máquina, etc.) cada día son más las aplicaciones de la visión artificial y en un mayor número de áreas.

2.3.1 Campos de aplicación de la visión por computador.

Hay múltiples campos en los que se pueden aplicar los avances en la visión por computador [4]:

- **Navegación en robótica:** En este caso se recurre a técnicas de visión estereoscópicas para poder reconstruir las escenas en 3D. Se combinan con módulos de reconocimiento 3D para identificar las distintas partes de la escena, ya sean determinados objetos, personas, etc. que deban ser evitados o a los que se deba dirigir. La visión por computador aplicada en esta área ha permitido que los robots tenga un mayor aspecto humanoide.

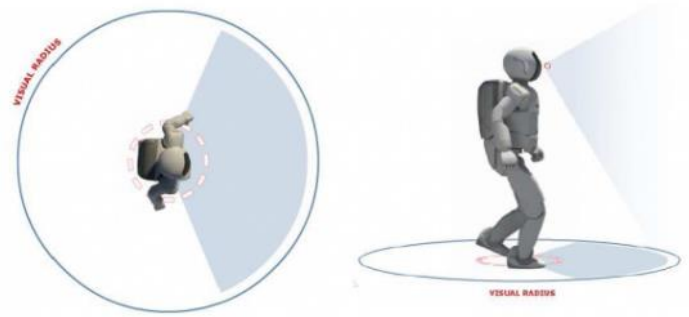


Ilustración 7. Alcance de la visión de un robot.

- **Biología, geología y meteorología:** Se puede aplicar tanto a nivel microscópico como a nivel macroscópico.

  En cuanto a imágenes microscópicas se trabaja con ellas para, por ejemplo, la identificación de distintos microorganismos o el número de ellos que hay presentes mediante distintas técnicas como la binarización y apoyándose en las características propias de cada microorganismo para su correcta identificación (tamaño, color, forma, etc.).

  Por otro lado, en las imágenes macroscópicas puede aplicarse a la identificación de determinados tipos de texturas en vegetales o en el crecimiento de ciertas especies. En campos como en la geología nos permite conocer si ha habido movimientos de terreno comparando dos imágenes. En este tipo de aplicaciones es importante que la iluminación se mantenga constante para evitar malos resultados.

- **Medicina:** En el campo de la medicina hay numerosas aplicaciones relacionadas con el diagnóstico de enfermedades y dolencias a través de radiografías, resonancias o tomografías. Se trata de un área extremadamente amplio.

- **Identificación de construcciones, infraestructuras y objetos en escenas de exterior:** Partiendo de imágenes aéreas o de satélites se puede identificar las distintas regiones de una zona, así como construcciones (edificios) o infraestructuras (carreteras, puentes, etc.) mediante técnicas de tratamiento de imagen como extracción de bordes o contornos o segmentación de regiones.

- **Fotointerpretación:** Es la ciencia que analiza las imágenes por parte de un experto para extraer la información relevante y de interés. Por ejemplo, situar las construcciones existentes en una imagen de un satélite en cierta área. Estas imágenes pueden ser tratadas con las técnicas necesarias para mejorar la calidad. Cuanto mayor sea la calidad inicial de la imagen, mejor será el resultado.

- **Y otras como:** Reconocimiento y clasificación, inspección y control de calidad, o cartografía.

2.3.2 Componentes de un sistema de visión artificial.

En el sistema de visión artificial entran en juego tres variables principales de los cuales dependerá el resultado final: La **iluminación** de los objetos a estudiar, la **cámara** y la **óptica** a través de la cual obtendremos la imagen y del **sistema de procesamiento** de imágenes (tanto el hardware; soporte físico, y software; los distintos algoritmos para la resolución del problema) [5].

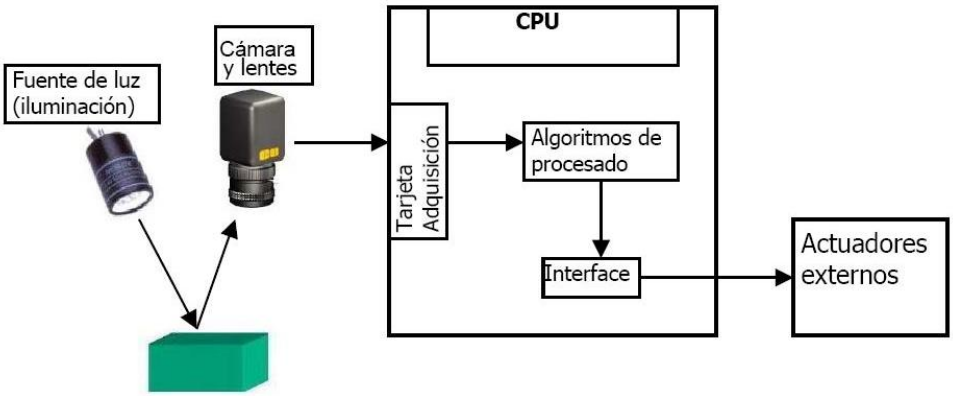


Ilustración 8. Componentes de un sistema de visión.

- **Iluminación**: En un sistema de percepción, las condiciones de iluminación del objeto o área a estudiar juegan un papel muy importante y una gran parte del resultado final depende de ella. A pesar de que tanto la iluminación como el procesamiento de la imagen tienen gran culpa del resultado final obtenido, se le suele dar a este último factor bastante mayor peso e importancia en el desarrollo de la aplicación, descuidando frecuentemente el papel que juega la iluminación. Siempre que sea posible se deben controlar las condiciones de iluminación para simplificar al máximo el análisis y la posterior interpretación de la imagen. Además, la iluminación existente alrededor del objeto a estudiar no solo ser suficiente, generando situaciones de escaso contraste y sombras.

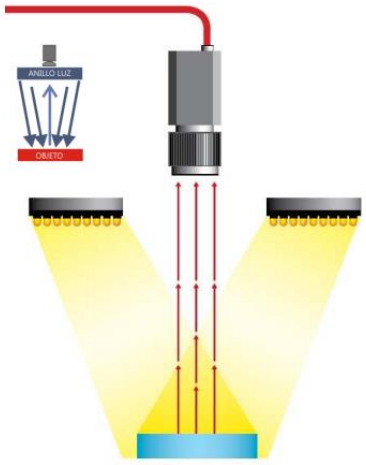


Ilustración 9.Iluminación frontal.

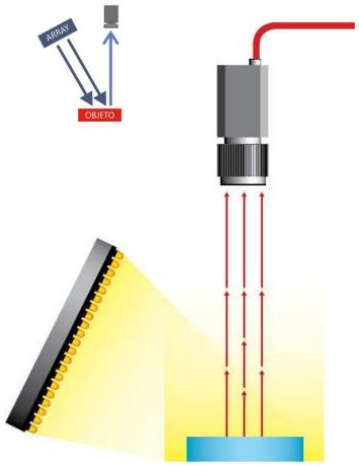


Ilustración 10. Iluminación lateral.

En la actualidad se dispone de un amplio abanico de fuente de luz y configuraciones que permiten destacar las características de interés de la imagen. También hay aplicaciones que trabajan en un espectro invisible al ojo humano como es el caso de los rayos X, los infrarrojos o los rayos ultravioleta útiles en algunos campos.

- **Cámara y óptica**: Las cámaras son las encargadas de captar la imagen a través de la óptica además de transmitirla al computador ya sea como señal analógica o digital. Las cámaras están dotadas de unos fotorreceptores a través de los cuales capta la imagen, pero anteriormente a eso, la óptica se ha encargado de hacer una correcta proyección de la imagen en ellos.

  La continua evolución de las cámaras ha permitido mejorar en gran nivel la calidad de las imágenes obtenidas a través de ellas, lo que permite un mejor estudio de las imágenes y resultados. Opciones que hace uno años eran inviables a nivel doméstico se han vuelto totalmente accesibles.

- **Sistema de procesamiento de imágenes**:

  Hardware: Los factores más influyentes en el impulso del campo de la visión artificial son el fuerte aumento de la potencia computacional de los dispositivos electrónicos como el abaratamiento de estos. Donde antes se necesitaba un

hardware especializado para el tratamiento de imágenes, ahora bata con un computador personal de bajo coste.
En la actualidad ya han aparecido cámaras con microprocesadores incluidos. Permite que, al realizar una captura, esta quede guardada en forma digital en una zona de la memoria a la que pueda ser accedida desde un computador para su posterior tratamiento.
En ciertas aplicaciones en las que se necesita capturar objetos que se mueven rápidamente, las cámaras incluyen el llamado trigger que permite tener una calidad óptima del objeto en cuestión.

En cuanto al computador, debe ser capaz de tratar la gran cantidad de datos que forman las imágenes a una velocidad suficientemente rápida para procesar los datos a tiempo real, algo muy complicado puesto que puede recibir varias decenas de imágenes por segundo.
Aunque hay aplicaciones que, por su enorme complejidad, necesitan de computadores especializados de elevado coste para poder procesar la información, gracias al gran desarrollo de las prestaciones de los procesadores la gran mayoría de las aplicaciones pueden solucionarse con un computador convencional o con cámaras inteligentes basadas en DSP (Procesador Digital de Señales).

Existen sistemas integrados compactos que incorporan una cámara con su óptica y el sistema de procesamiento completo formando parte de un solo sistema. Se trata de sistemas pensados para clientes finales y por ello son sistemas relativamente fáciles de configurar y de adaptar a cada aplicación. Además, al carecer de discos duros son muy robustos ante vibraciones o choques y pueden emplearse en aplicaciones industriales especialmente hostiles. Estos sistemas incluyen además funcionalidades para la entrada y salida que permiten la comunicación con PLC, relés y robots. No obstante, el procesador de estos sistemas es de menores prestaciones que el de un PC por lo que deberá extremarse en este caso el cuidado para obtener imágenes de buen contraste y evitar de esta forma cualquier procesamiento adicional.

Software: En cada caso dependerá del problema a resolver. Tareas como comprobación de marcas impresas o etiquetas, detección de defectos, inspección dimensional, clasificación,... requieren de distintos algoritmos para su resolución.

Es complicado conocer a priori que algoritmos y metodología será la más adecuada para resolver cada problema, así como el tiempo de proceso. La experiencia y las pruebas sirven de gran ayuda para diseñar la mejor solución ya que no hay una metodología modelo a seguir para la resolución de estos problemas.

Para su resolución existen distintas librerías de procesamiento de imágenes con distintos tipos de funciones como filtrados, extracción de contornos, regiones, análisis de características, etc.

La tecnología, cada vez más presente, y cada vez más barata, nos puede ayudar de múltiples maneras para mejorar la seguridad en nuestro coche. Una cámara digital es un dispositivo pequeño y económico que puede montarse en un coche sin grandes complicaciones. Pero más importante que esto es el cerebro del sistema, el procesador, y el software, que se encargan de reconocer adecuadamente y en toda circunstancia lo que capta la cámara.

2.3.3 Aplicaciones de la visión por computador en seguridad vial.

A continuación se enumerarán y explicarán algunas aplicaciones de la visión artificial más novedosas en el campo de la seguridad vial.

2.3.3.1 Cámaras de reconocimiento facial.

Tres de cada cuatro accidentes de tráfico son debido al factor humano, y algunos de ellos por temas de fatiga, cansancio o fatiga del conductor. Por ello se están desarrollando sistemas de visión capaces de reconocer cuando el conductor padece alguno de estos efectos y evitar males mayores [6].

A través de una cámara colocada enfocando la cara del conductor y a un sistema de reconocimiento facial, es capaz de reconocer si sufre cansancio, fatiga, sueño o falta de concentración y generar alguna respuesta para evitar un accidente.

La cámara se encarga de supervisar los ojos para ver si el parpadeo es normal, o si el parpadeo indica sueño, pero también es capaz de ver si el conductor mira al frente, a la carretera, o si desvía la mirada a otra parte, fuera de la carretera, y está dejando de atender a la circulación, mirando cualquier otra cosa como un teléfono, la radio, o el paisaje por la ventanilla.

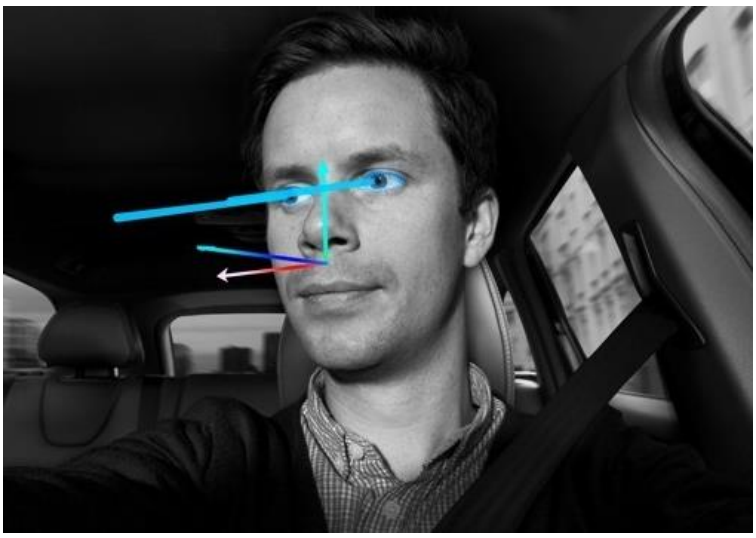


Ilustración 11. Aplicación cámara reconocimiento facial.

Realiza un reconocimiento facial completo, no solo analiza los ojos, para ser capaz de detectar cuanto antes los síntomas de cansancio. Es capaz de reconocer bostezos y otras expresiones faciales que muestren cansancio y fatiga. No solo detecta síntomas de cansancio sino también estados de ánimo como estrés, nervios o cólera, síntomas que pueden ser peligrosos a la hora de conducir.

Actúan alertando al conductor de la situación mediante una alarma, luces rojas parpadeantes, o mensaje de texto en el cuadro de instrumentos, e instándole a no distraerse, parar para descansar, dormir o calmarse. Que el conductor haga caso es voluntario, pero una cosa es que no se dé cuenta de que le está pasando, y otra diferente que, avisado del problema, insista en seguir conduciendo así.
Distintos fabricantes como Volvo, PSA Peugeot Citroën, Toyota o BMW están investigando este tipo de tecnología.

2.3.3.2 Monitorización de ocupantes en el interior de vehículos.

Es un sistema de monitorización de ocupantes dentro de los vehículos mediante el uso de visión computacional. Este sistema cuenta el número de ocupantes del vehículo, distinguiéndolos entre adultos, niños y bebes y, en caso de accidente, graba el último minuto posterior a él. Además conecta automáticamente con el servicio de emergencias avisándole de que ha ocurrido un accidente [7].

Para el conteo de ocupantes utiliza el algoritmo de Viola-Jones para la detección de caras.

Debido a la posible distorsión de la imagen se utilizan técnicas de calibración y rectificación como paso previo a la extracción de información de la imagen.

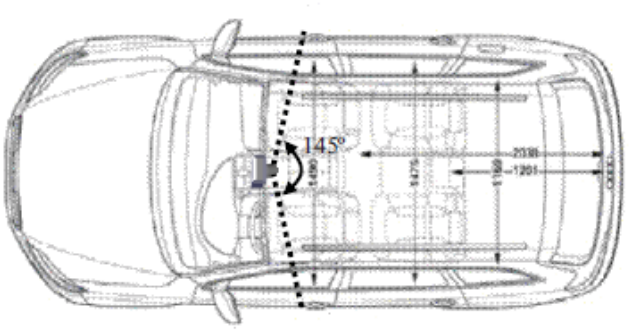


Ilustración 12. Esquema del vehículo y colocación de la cámara en su interior así como el ángulo de visión encubierto.

Una de las claves de esta aplicación es el ángulo de visión máximo del montaje cámara más lente tipo “ojo de pez”.

2.3.3.3 LDW y LKS.

LDW son las siglas de “Lane Departure Warning” y se trata de un detector de cambio de carril. El sistema de seguridad detecta la variación de trayectoria del vehículo, lo interpreta, y en caso de ser un cambio involuntario avisa al conductor para evitar cualquier tipo de accidente [8].

Para detectar las marcas viales utiliza sensores que pueden ser de tipo cámara y suelen ir situados en el parabrisas o en los bajos del vehículo en caso de que utilice infrarrojos

para la detección de las líneas. Realiza la lectura y distingue entre marcas continuas y discontinuas y manda esta información a la centralita del LDW.

Para decidir si la maniobra es a propósito o no, el sistema estudia distintas variables como el grado del giro del volante o la activación o no de los intermitentes y de esta forma conocer si el conductor está realizando un movimiento voluntario o ha perdido la trayectoria del carril. En este último caso, el sistema emitiría una alarma tanto visual como sonora.

De forma adicional, hay modelos en los que el sistema hace vibrar el volante o incluso el asiento para alertar al conductor del riesgo de pérdida de control del vehículo.

Ilustración 13. Sistema LDW con sensor tipo cámara.

El **LKS** (de *Lane Keeping System*) es la evolución del LDW. El LDW se limita a detectar el problema y avisar al conductor mientras que el LKS, además intenta mantener el coche en el carril actuando sobre la columna de la dirección del vehículo para girar sus ruedas.

También pueden avisar al conductor cuando no detectan sus manos sobre el volante.

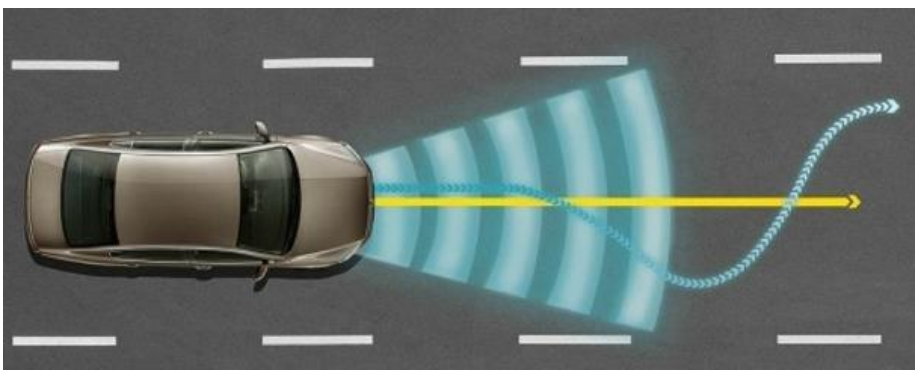


Ilustración 14. Sistema LKS.

A la práctica, la mayoría de estos sistemas pecan de un problema común, y es que no siempre son capaces de detectar la desviación de la trayectoria, bien porque las marcas no sean visibles de forma suficiente, bien porque el sensor no termine de reconocerlas.

2.3.3.4 Reconocimiento de señales de velocidad

Al igual que en el caso anterior, se equipa el vehículo con una cámara de alta resolución colocada en la parte alta del parabrisas centrada y vigila los márgenes de la carretera para leer las señales que hay en ellas. Reconoce varios tipos de señales como las de velocidad máxima, prohibido adelantar, fin de velocidad máxima y fin de prohibido adelantar [9].

La pantalla digital del cuadro muestra en todo momento la velocidad límite del tramo en el que nos encontremos y se actualiza a tiempo real con las nuevas señales.

La cámara destinada al reconocimiento de estas señales puede ser la misma destinada al reconocimiento de carriles.

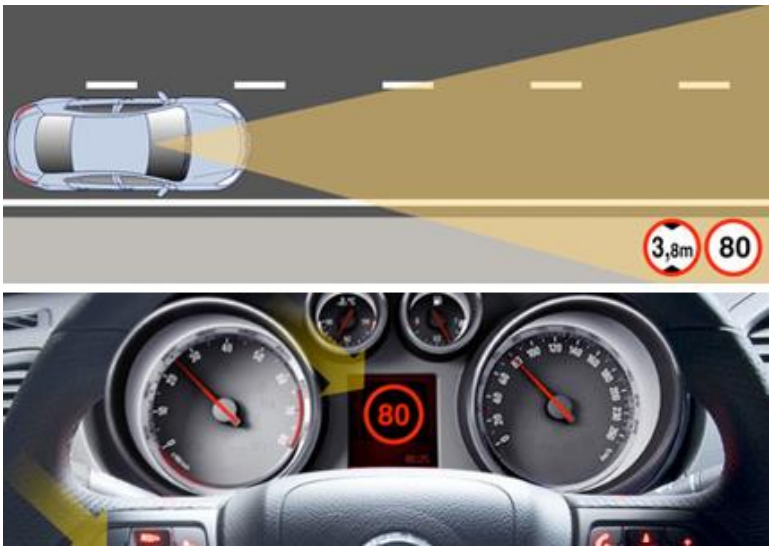


Ilustración 15. Reconocimiento de señales de velocidad.

Algunos modelos como el Ford S-Maz incluyen además una respuesta en función a la señal que detectan limitando la velocidad máxima que el coche puede alcanzar.

2.3.3.5 Sistema ContiGuard

En este caso, el sistema se compone de dos cámaras de alta resolución capaces de reconstruir una imagen estereográfica (crear la ilusión de profundidad) también colocadas por encima del parabrisas. Es capaz de reconocer un gran número de obstáculos próximos y activar el sistema de frenado del vehículo [10].

Mediante la relación entre la reflexión de la luz en los objetos (para conocer su distancia) y la velocidad del vehículo decide si hay riesgo de colisión con el obstáculo de delante, y en ese caso activa el sistema de frenado. Funciona con distintos tipos de objetos como obstáculos en carretera, animales cruzando las vías, ciclistas o peatones.

Las cámaras tienen un alcance de 60 metros y una precisión de, alrededor de los 20 centímetros. Funcionan también en condiciones adversas climáticas, aumentando aún más las precauciones.

Ilustración 16. Sistema ContiGuard.

2.3.3.6 Cámaras para ver lo que el conductor no ve.

Es un sistema de visión con cámaras que pueden estar instaladas o en la parte trasera o en la parte delantera del vehículo [11].

Las cámaras instaladas en la parte posterior permiten ver, a través de la pantalla del sistema de navegación del coche, si hay algún obstáculo detrás a la hora de dar marcha atrás. Ya sea un obstáculo que pueda dañar el coche o niños pequeños a los que sea imposible ver a partir de los retrovisores.

Por otro lado, si las cámaras están instaladas en la parte delantera del coche, deben estar en las esquinas del parachoques delantero. Estas cámaras permiten ver hacia ambos lados en los cruces con poca visibilidad en los que el conductor está a cierta distancia de la parte más adelantada del coche.

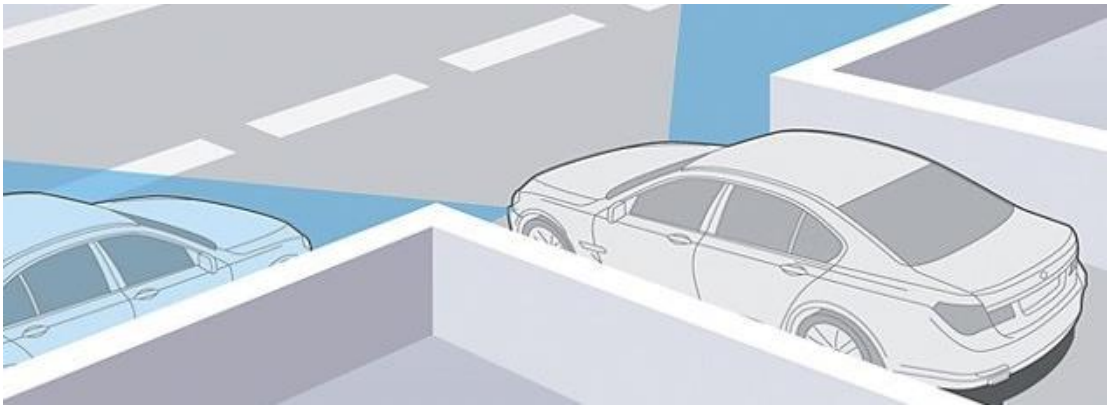


Ilustración 17. Cámara de visión lateral para los parachoques delanteros.

2.3.3.7 Sistemas ANPR.

Estas iniciales significan "Automatic number plate recognition". Reconocen automáticamente la matrícula de los vehículos [12].

La tecnología permite la vigilancia masiva de vehículos mediante el uso de un software óptico que escanea las imágenes y lee las matrículas automáticamente. De esta forma evita que haya que mecanografiar a mano todas y cada una de las matrículas.

Ilustración 18. Aplicación sistemas ANPR.

Estas técnicas son utilizadas por las diversas fuerzas de policía y como método de recaudación electrónica de peaje en las autopistas de pago o para vigilar la entrada y salida de aparcamientos privados, entre otros ejemplos.

2.3.3.8 Coche autónomo de Google.

El coche autónomo de Google se trata de un coche que se conduce solo, por todo tipo de vías, realizando todo tipo de acciones, y sin la intervención del conductor en ningún momento. Combina un computador, diferentes sensores y automatismos para que se mueva solo [13].

Se conduce por sí solo gracias a la combinación de distintas tecnologías como reconocimiento de carriles, señales de tráfico, semáforos y cruces. Distingue a otros vehículos, ciclistas y peatones. Controla la distancia de seguridad con el vehículo de delante y toma decisiones para evitar accidentes. Todo lo que haría un conductor responsable cualquiera.

No se puede decir que un elemento del conjunto de tecnologías que incorpora sea más importante que otro ya que es la combinación y complementación de todos ellos los que permiten que el sistema funcione con garantías y seguridad. La mayoría además ya existen hoy en día y se implementan de manera independiente en diferentes coches.

El elemento más importante del vehículo es el LIDAR (Laser Imaging Detection and Ranging) es decir un dispositivo que detecta objetos y mide la distancia hasta ellos mediante rayos de luz, concretamente haces láser. Gracias al LIDAR se construye una imagen tridimensional alrededor del coche, con todo tipo de objetos posicionados. No obstante, también precisa de una **cámara** encargada de **reconocer las señales de tráfico, los semáforos y las líneas de la carretera**.

Ilustración 19. Cámara del coche autónomo de Google.

2.3.3.9 Luces del coche que iluminarán donde mire el piloto.

Se trata de un avance desarrollado por la compañía Opel relacionado con el uso más adaptado de la iluminación [14].

Esta nueva tecnología presenta como principal novedad la prestación del seguimiento ocular. La iluminación de los faros del coche variará su dirección en función de donde mire el conductor en cada momento. Para conseguirlo se instala una cámara enfocando la cara del conductor y se le dota de un software de seguimiento de la mirada.

Ilustración 20. Escaneo de los ojos.

Este avance que se atribuye Opel simplifica de forma versátil el sistema. Para ello, ha sido necesario optimizar de los parámetros de funcionamiento de la cámara usada y trabajar con el algoritmo de seguimiento ocular.

Ilustración 21. Iluminación del coche en función de los ojos del conductor.

La cámara está equipada con sensores infrarrojos periféricos y fotodiodos centrales que permiten escanear los ojos del conductor más de 50 veces por segundo incluso en condiciones de menos luz.

Otro punto importante, además, es la de conseguir que esa detección de movimiento se transmita lo más rápido posible a los faros del coche consiguiendo una sensación de movimiento instantáneo en coordinación con la mirada.

2.3.4 Aplicaciones en seguridad vial para Android.

A continuación se recogen algunos ejemplos de aplicaciones en el campo de la seguridad vial para Android:

2.3.4.1 iOnRoad.

Se trata de una aplicación para Android que avisa de posibles colisiones. Aprovecha los recursos de los SmartPhones para optimizar la conducción en tiempo real. La App se sirve de la cámara y los sensores GPS incorporados en el SmartPhone para detectar cualquier vehículo que tengamos delante y alertar al conductor de posibles peligros. El radar visual informa en tiempo real de cualquier objeto que se halle frente al conductor y calcula la velocidad actual del vehículo basándose en los sensores incorporados. Cuando el vehículo se aproxima a un peligro se dispara una señal audiovisual que alerta de una posible colisión, lo que permite al conductor frenar a tiempo [15].

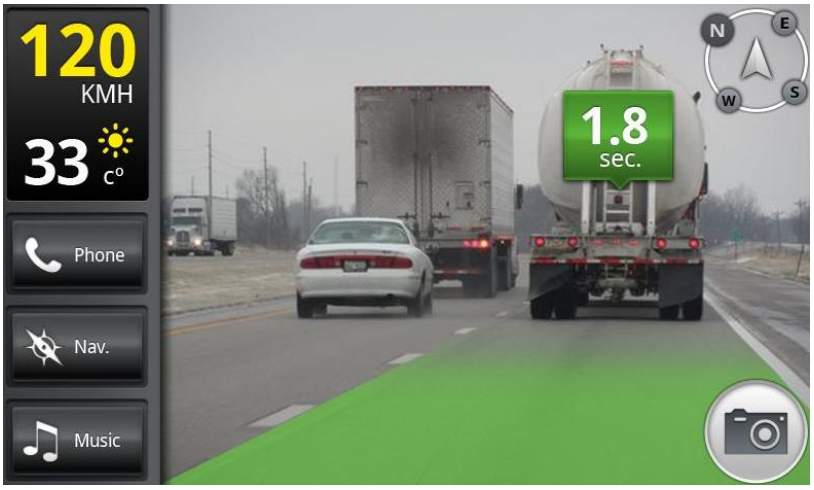


Ilustración 22. Aplicación iOnRoad.

2.3.4.2 Drivea.

Se trata de una aplicación similar a la anterior que controla distintos riesgos como las trayectorias que pueden provocar un accidente, si nos salimos de un carril o si sobrepasamos la velocidad límite [16].

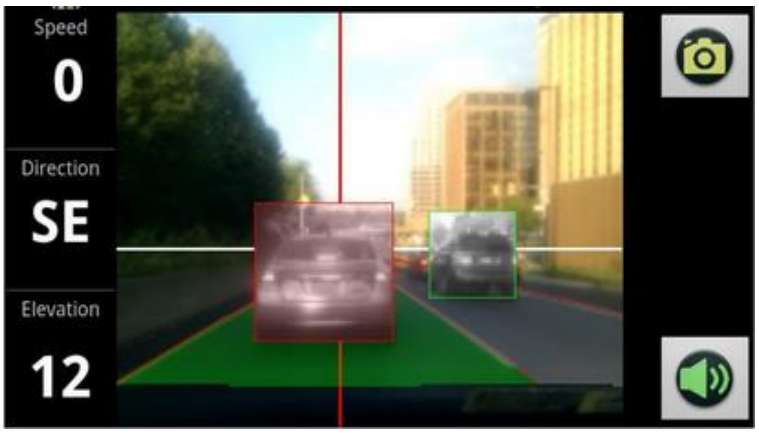


Ilustración 23. Aplicación Drivea.

2.3.4.3 Reconocimiento automático de matrícula en Android.

Se trata de una aplicación encaminada a la detección de matrículas argentinas a través de la cámara de cualquier dispositivo inteligente sobre plataforma Android. Funciona como los sistemas ANPR (Automatic Number Plate Recognition), anteriormente explicados, adaptado a Android. Además se ha realizado con la biblioteca OpenCV.

La aplicación pasa por distintas fases para poder extraer el número de la matrícula: Escaneo óptico, localización y segmentación, pre-procesamiento, extracción de características y por último el reconocimiento y el pos-procesamiento [17].

Ilustración 24. Etapas de un reconocedor óptico de caracteres.

2.4 OpenCV

OpenCV (**Open Source Computer Vision**) es una librería software open-source de visión artificial y machine learning que provee una infraestructura para aplicaciones de visión artificial originalmente desarrollada por Intel [1].

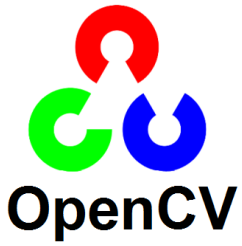


Ilustración 25. Logo OpenCV.

Tiene una licencia BSD (código fuente muy cercano al dominio público), lo que permite utilizar y modificar el código, tiene una comunidad de más de 47000 personas y más de 7 millones de descargadas. Es una librería muy usada a nivel comercial, desde Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, Applied Minds, VideoSurf, Zeitera, etc.

La librería tiene más de 2500 algoritmos, que incluye algoritmos de machine learning y de visión artificial para usar. Estos algoritmos permiten identificar objetos, caras, clasificar acciones humanas en vídeo, hacer tracking de movimientos de objetos, extraer modelos 3D, encontrar imágenes similares, etc.

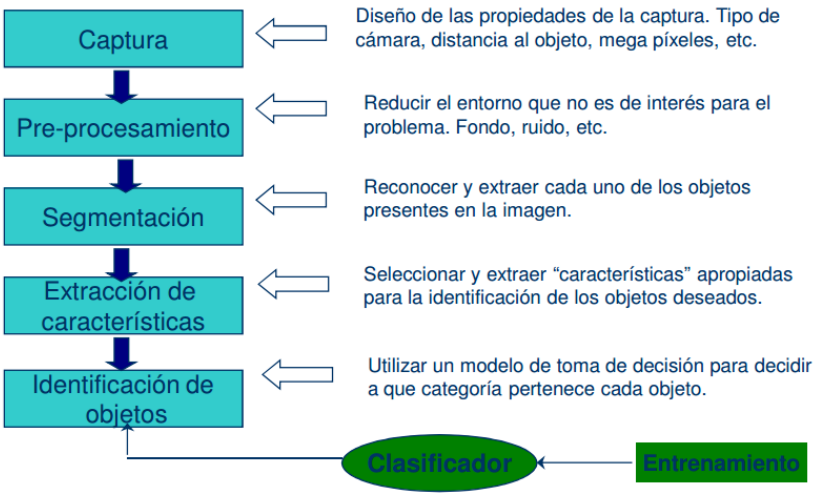


Ilustración 26. Etapas del procesamiento de imágenes.

OpenCV está escrito en C++, tiene interfaces en C++, C, Python, Java y MATLAB y funciona en Windows, Linux, Android y Mac OS [18].

El proyecto pretende proporcionar un entorno de desarrollo fácil de utilizar. Esto se ha logrado, realizando su programación en código Java.

2.5 Proyectos del LSI

Paralelamente, desde el Laboratorio de Sistemas Inteligentes (LSI) de la Universidad Carlos III de Madrid se está trabajando en otros proyectos relacionados con este mismo tema de la seguridad vial. Todos ellos desarrollados en Android.

2.5.1 Biblioteca para la localización de peatones en dispositivos Android.

Consiste en la creación de una librería para Android que proporciona datos útiles orientados al trabajo con la cámara del dispositivo y el cálculo de distancias y la aplicación de dicha librería a un software de detección de peatones para calcular la distancia del dispositivo a dicho objeto [16].

Ilustración 27. Biblioteca para el cálculo de distancia de personas.

2.5.2 Aplicación Android para la detección de señales de tráfico.

Mediante la cámara de los SmartPhones de Android Identifica y distingue los distintos tipos de señales que se pueden encontrar en la carretera [19].

Mediante identificación de formas y colores es capaz de extraer las señales presentes en la imagen y comparándolos con una base de datos anteriormente introducida en la aplicación es capaz de distinguir el tipo de señal.

Ilustración 28. App Android para la detección de señales de tráfico.

2.5.3 Adquisición remota de datos mediante el sistema OBD-II y dispositivo móvil.

Se encarga de recoger distintos datos del vehículo en el que nos encontramos a través del SmartPhone y suministrarlos a la red. Flexibiliza el tipo de datos, la frecuencia con la que se recogen y el procesado requerido para permitir la compatibilidad con otras aplicaciones y servicios [20].

Para adquirir los datos del sistema del vehículo lo hace a través de un conectar OBD-II, algo compatible con la mayoría de los vehículos de forma que abarca una gran parte del mercado.

Ilustración 29. Elementos para la adquisición de datos.

2.5.4 Implementación de algoritmos de visión por computador en plataforma Android.

Desarrolla por una parte, una aplicación para dispositivos móviles Android que pueda recoger las distintas subaplicaciones desarrolladas a posterior en el Laboratorio de Sistemas Inteligentes (LSI). Por otro lado, dos subaplicaciones; una para detectar peatones y otra para detectar vehículos [21].

Ilustración 30. App para la detección de vehículos.

3. Software

3.1 Eclipse

En un primer momento se barajó la posibilidad de trabajar con el software Android Studio ya que con el tiempo se convertirá en el software más extendido para el desarrollo de aplicaciones Android. No obstante, se hubiese trabajado con su primera versión estable que, al estar en su fase inicial es susceptible de introducir más cambios que puedan provocar inestabilidad entre proyectos de diferentes versiones.

Por este motivo, el entorno de desarrollo elegido para el desarrollo de la aplicación es la versión Luna de Eclipse.

Ilustración 31. Logo Eclipse.

Eclipse es una plataforma de desarrollo de código abierto basada en Java. Por si misma, es simplemente un marco de trabajo y un conjunto de servicios para la construcción del entorno de desarrollo de los componentes de entrada. Eclipse tiene un conjunto de complementos, incluidas las Herramientas de Desarrollo de Java (JDT) [22].

Para poder trabajar con este entorno aplicado a Android es necesario hacer distintas configuraciones e instalaciones en las que hay que incluir:

- Eclipse Luna.
- El SDK de Android todas las herramientas de desarrollo y emuladores.
- La versión de ADT (Android Developer tools) preinstalada en Eclipse
- Los paquetes y la máquina virtual de Android 4.2 (Jelly Bean).

4. Aplicación detección de las líneas de la carretera.

Para crear esta aplicación partimos de la ya creada anteriormente que contendrá todas las subaplicaciones.

4.1 Aplicación de partida.

Se trata de una aplicación creada por el laboratorio de investigación de la Universidad Carlos III de Madrid que recibe el nombre de LSI (Intelligent System Lab).

Ilustración 32. Pantalla principal aplicación LSI.

En su pantalla principal podemos observar distintos botones:

- "Aplicaciones LSI" desde el cual podemos elegir entre las distintas subaplicaciones incluidas y desarrolladas en la App.

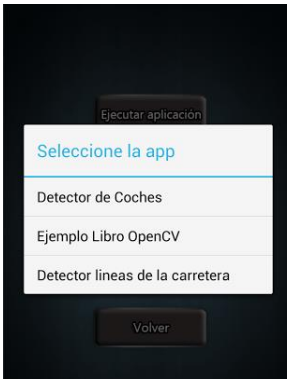


Ilustración 33. Pantalla que redirige a cada una de las aplicaciones.

En este caso, la aplicación de interés es la llamada “Detector de líneas de la carretera”.

- "Ejemplos” que nos muestra distintas aplicaciones para la cámara de nuestro dispositivo Android descargadas directamente desde la página web de OpenCV y que nos permite tener un primer acercamiento con la librería.

- “¿Quiénes somos?” donde encontraremos la información del primer autor de la aplicación y el tutor y director del proyecto.

Ilustración 34. Ejemplos OpenCV.

Ilustración 35. Desarrolladores aplicación.

- Por último el botón “Universidad Carlos III de Madrid” nos redirige a la página principal de la universidad.

4.2 Desarrollo de la aplicación.

El objetivo final es conseguir distinguir las distintas líneas presentes en la imagen, resaltarlas y diferenciarlas entre continuas y discontinuas. Para conseguirlo, se trata cada fotograma o frame de la imagen con distintos filtros y algoritmos para obtener el resultado deseado.

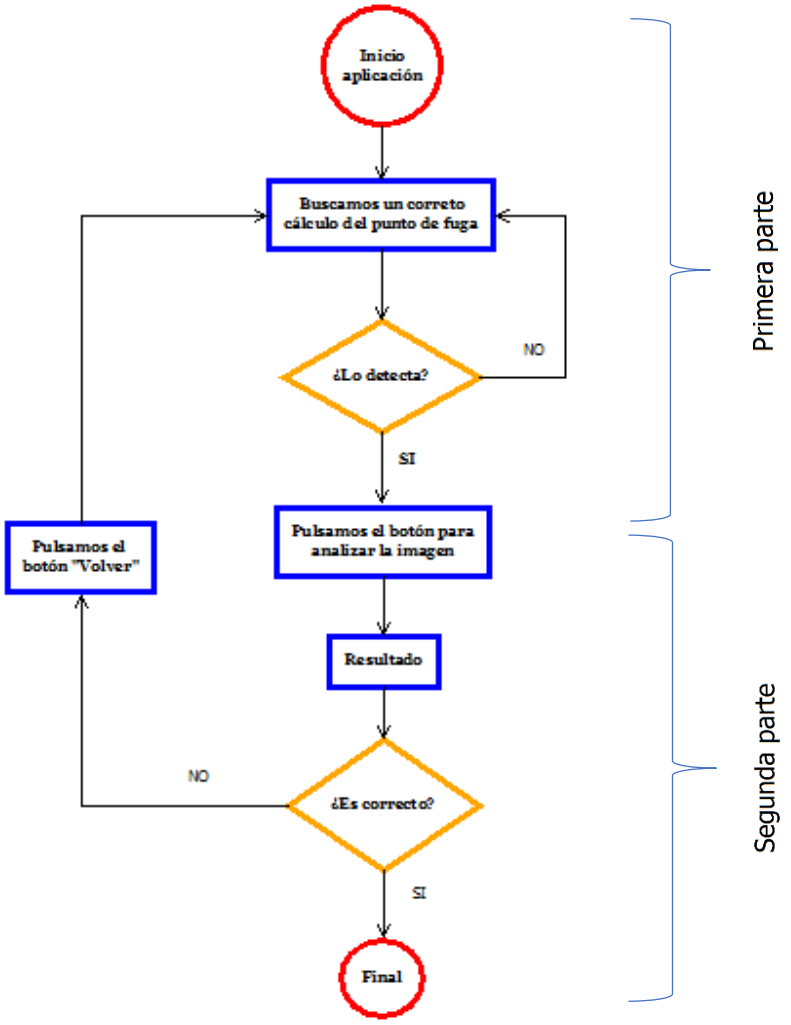


Ilustración 36. Estructura de la aplicación.

Se pueden diferenciar dos partes de código claras, una en la que se trata la imagen hasta obtener lo que llamaremos “vista de pájaro” y otra parte en la que se analizará esa imagen para conseguir el resultado final. Para dar paso de una parte a otra se hará con un botón ya que, al estar la aplicación en desarrollo nos permitirá identificar mejor los errores y corregirlos.

4.2.1 Búsqueda de la vista de pájaro.

El objetivo de buscar una "vista de pájaro" de la imagen original y a partir de ella analizarla es que nos proporcionara una vista proporcional de toda la carretera. En la imagen original, las zonas más cercanas a la cámara son de mayor tamaño y se aprecian mejor lo detalles mientras que por otro lado, las zonas lejanas resultan más difíciles de apreciar. Por ejemplo, observemos la siguiente imagen:

Ilustración 37. Posible situación a resolver.

Si queremos analizar una parte significativa de la carretera para determinar la continuidad de las líneas debemos coger una parte importante de ella, no vale únicamente con las zonas más cercanas. A medida que profundizamos en la imagen podemos apreciar como el hueco entre las líneas que dividen ambos carriles son cada vez más difíciles de ver, y esto a la hora de analizar la imagen puede hacer que obtengamos falsos negativos, es decir, que nos detecte la línea como continua.

Con la vista de pájaro lo que conseguiremos es que los huecos de las zonas más profundas de la imagen sean mayores y que estén proporcionadas todas las partes.

La estructura que se ha seguido para un correcto procesamiento de la imagen es la siguiente:

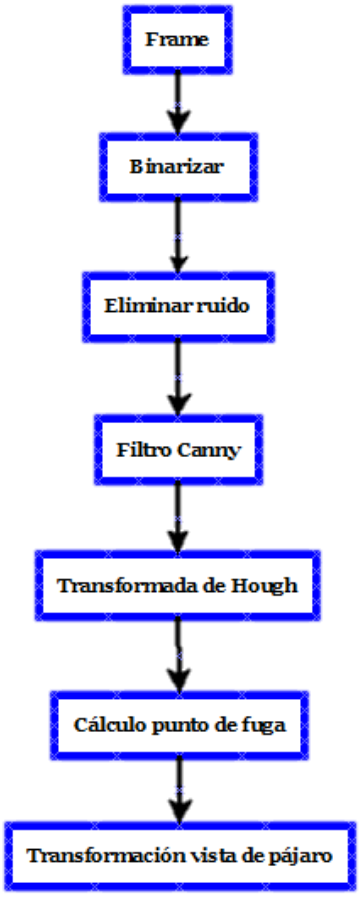


Ilustración 38. Estructura primera parte del código.

Partimos de la imagen de una carretera en la que sus principales características (color y líneas) se distinguen fácilmente:

Ilustración 39. Punto de partida.

- **Binarización de la imagen:**

En un primer momento binarizamos la imagen. Binarizar una imagen consiste en un proceso de reducción de la información de la misma, en la que sólo persisten dos valores: verdadero y falso. En una imagen digital, estos valores, verdadero y falso, pueden representarse por los valores 0 y 1 o, más frecuentemente, por los colores negro (valor de gris 0) y blanco (valor de gris 255).

En el proceso y análisis de imagen, la binarización se emplea para separar las regiones u objetos de interés en una imagen del resto y para crear máscaras sobre regiones. Esto es exactamente lo que buscamos, destacar sobre el resto las líneas blancas de la carretera ya que, como podemos observar, todas estas líneas se cortan en un mismo punto en el horizonte (punto que utilizaremos como referencia para hacer la transformación a vista de pájaro). Obtenemos esta imagen:

Ilustración 40. Resultado binarizado.

Como podemos observar en la anterior imagen, la única información que tenemos es la de las líneas de la carretera además de algo de "ruido" (características de la imagen que no aportan información útil y que puede provocar errores en su análisis). No obstante, la aparición de más o menos "ruido" depende de cada caso y de los niveles de gris presentes en la imagen.

- **Eliminación de "ruido":**

Una característica que estará siempre presente en este tipo de imágenes es, que a partir de cierta altura, la información que encontremos en ella será totalmente inútil ya que la carretera se corta en el horizonte a cierta altura. Por ello se puede eliminar la información que obtengamos a partir del horizonte hacia arriba:

Ilustración 41. Resultado binarizado con eliminación de ruido.

De esta forma, los procesos posteriores estarán menos expuestos a posibles errores.

- **Filtro Canny:**

El algoritmo Canny se trata de un método relacionado con la detección de bordes basado en el uso de la primera derivada, la que es usada por que toma el valor de cero en todas las regiones donde no varía la intensidad y tiene un valor constante en toda la transición de intensidad. Por tanto un cambio de intensidad se manifiesta como un cambio brusco en la primera derivada, característica que es usada para detectar un borde.

El algoritmo de Canny consiste en tres grandes pasos:

1. Obtención del gradiente: en este paso se calcula la magnitud y orientación del vector gradiente en cada píxel.
2. Supresión no máxima: en este paso se logra el adelgazamiento del ancho de los bordes, obtenidos con el gradiente, hasta lograr bordes de un píxel de ancho.

3. Histéresis de umbral: en este paso se aplica una función de histéresis basada en dos umbrales; con este proceso se pretende reducir la posibilidad de aparición de contornos falsos.

Para la **obtención del gradiente**, lo primero que se realiza es la aplicación de un filtro gaussiano a la imagen original con el objetivo de suavizar la imagen y tratar de eliminar el posible ruido existente. Sin embargo, se debe de tener cuidado de no realizar un suavizado excesivo, pues se podrían perder detalles de la imagen y provocar un pésimo resultado final. Este suavizado se obtiene promediando los valores de intensidad de los píxeles en el entorno de vecindad con una máscara de convolución de media cero y desviación estándar. Una vez que se suaviza la imagen, para cada píxel se obtiene la magnitud y módulo (orientación) del gradiente [23].

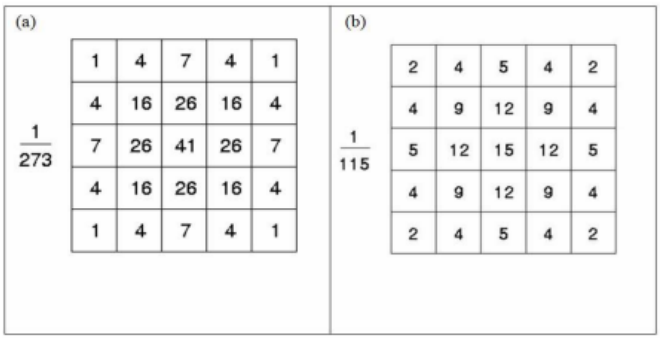


Ilustración 42. Máscaras que se pueden usar para realizar el filtrado gaussiano.

Para el **adelgazamiento de bordes** el procedimiento es el siguiente: se consideran cuatro direcciones identificadas por las orientaciones de 0º, 45º, 90º y 135º con respecto al eje horizontal. Para cada píxel se encuentra la dirección que mejor se aproxime a la dirección del ángulo de gradiente.

1. Suavizar la imagen I con H mediante un filtro gaussiano y obtener J como imagen de salida.

2. Para cada píxel (i, j), obtener la magnitud y orientación del gradiente basándose en las siguientes expresiones: El gradiente de una imagen f(x,y) en un punto (x,y) se define como un vector bidimensional dado por la ecuación:

$$G[f(x,y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} f(x,y) \\ \frac{\partial}{\partial y} f(x,y) \end{bmatrix}$$

Siendo un vector perpendicular al borde, donde el vector G apunta en la dirección de variación máxima de f en el punto (x,y) por unidad de distancia, con la magnitud y dirección dadas por:

$$|G| = \sqrt{G_x^2 + G_y^2} = |G_x| + |G_y| ,$$
$$\phi(x,y) = \tan^{-1} \frac{G_y}{G_x}$$

Donde:

- **Entrada**: imagen I máscara de convolución H, con media cero y desviación estándar σ .
- **Salida**: imagen Em de la magnitud del gradiente imagen Eo de la orientación del gradiente

3. Obtener Em a partir de la magnitud de gradiente y Eo a partir de la orientación, de acuerdo a las expresiones anteriores.

Por último, la **histéresis del umbral** La imagen obtenida en el paso anterior suele contener máximos locales creados por el ruido. Una solución para eliminar dicho ruido es la histéresis del umbral. El proceso consiste en tomar la imagen obtenida del paso anterior, tomar la orientación de los puntos de borde de la imagen y tomar dos umbrales, el primero más pequeño que el segundo. Para cada punto de la imagen se debe localizar el siguiente punto de borde no explorado que sea mayor al segundo umbral. A partir de dicho punto seguir las cadenas de máximos locales conectados en

ambas direcciones perpendiculares a la normal del borde siempre que sean mayo res al primer umbral. Así se marcan todos los puntos explorados y se almacena la lista de todos los puntos en el contorno conectado. Es así como en este paso se logra eliminar las uniones en forma de Y de los segmentos que confluyan en un punto.

Después de aplicar este filtro, la imagen que obtenemos es la siguiente:

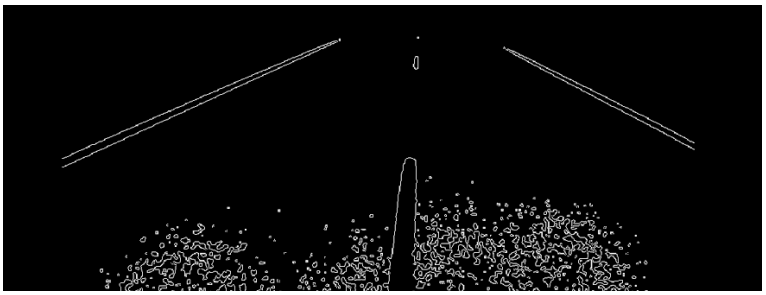


Ilustración 43. Resultado con filtro Canny.

- **Transformada de Hough:**

La transformada Hough es un algoritmo empleado en reconocimiento de patrones de una imagen, la cual permite encontrar formas como círculos, líneas, entre otras, dentro de la imagen. La versión utilizada para nuestro caso consiste en encontrar líneas, pero de acuerdo a la imagen y problema que se tenga se puede modificar para encontrar otro tipo de formas. El modo de funcionamiento es estadístico y de acuerdo a los puntos que se tengan se debe averiguar las posibles líneas en las que puede estar el punto, lo cual se logra por medio de una operación que es aplicada a cada línea en un rango determinado. La trasformada Hough utiliza dentro de su funcionamiento una representación paramétricas de forma geométrica, es decir, que si se tiene una recta, esta se representaría con los parámetros **r** y **θ,** donde **r** es la distancia entre a línea y el origen, y **θ** es el ángulo del vector desde el origen al punto más cercano. Por medio de la parametrización la ecuación de la recta se podría escribir de la siguiente manera [24].

$$r = x \cdot cos\theta + y \cdot sin\theta$$

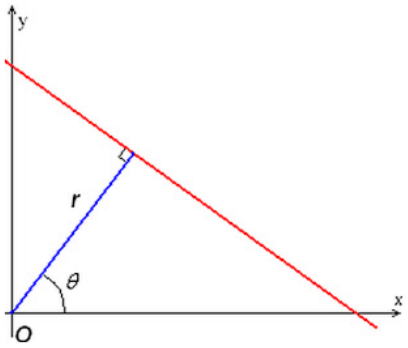


Ilustración 44. Ecuación y representación transformada de Hough para rectas.

Una de las características que posee la trasformada es que si se representan en un plano cartesiano la recta quedaría representada mediante las coordenadas (**r** ,**θ**), y el punto se representaría como una función senoidal. Por tal motivo si se tienes dos puntos, estos se simbolizarían por medio de dos senoides desfasadas alfa grados de acuerdo a las coordenadas de los puntos, pero si los dos puntos compartes la misma recta las dos senoides se terminaran cruzando cada 180 grados ya que la función senoidal representa el conjunto de las infinitas rectas que pasan por el punto, tal y como se observa en la siguiente figura:

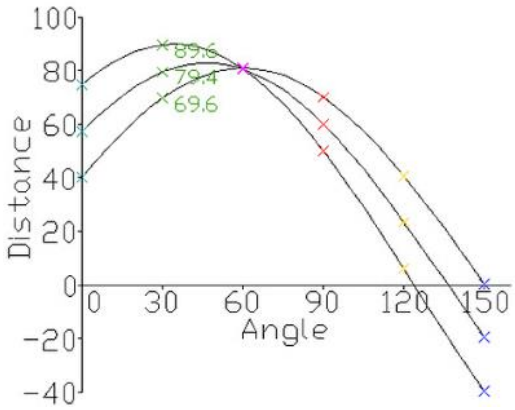


Ilustración 45. Representación senoidal de los puntos.

Una vez conocidos los parámetros que definen la transformada de Hough se puede aplicar a nuestra imagen. Puesto que se pueden generar líneas con distintos ángulos y para este caso solo nos interesan los que abarcan cierto ángulo (líneas cercanas a la vertical) se pueden descartar fácilmente el resto con la ayuda del parámetro θ.

Además, también es necesario distinguir las líneas de la izquierda de las de la derecha, también con el parámetro θ. Las líneas de la izquierda están inclinadas hacia la derecha y las de la derecha hacia la izquierda. Esta distinción será necesaria para el paso siguiente en el que se calculará en punto de fuga (corte entre las rectas de Hough).

El resultado que obtenemos después de este paso es el siguiente:

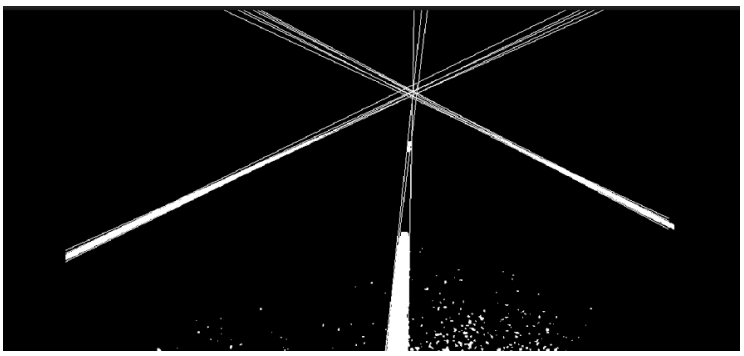


Ilustración 46. Resultado con la transformada de Hough.

- **Cálculo del punto de fuga.**

Una vez diferenciadas las rectas entre las que formas un ángulo u otro es sencillo calcular el punto de corte entre ellas a partir de las coordenadas de dos puntos de cada una de ellas. Estos puntos son de fácil acceso ya que son uno de los parámetros que caracteriza a la transformada de Hough.

Se calculan varios de estos puntos, es decir, el corte de una línea de la izquierda con varias de la derecha y así sucesivamente. Cuando tengamos un número significativo de cortes, se calcula la media de todos ellos para obtener un resultado más fiable. Este punto se guardará ya que será a partir del cual se construirá la vista de pájaro.

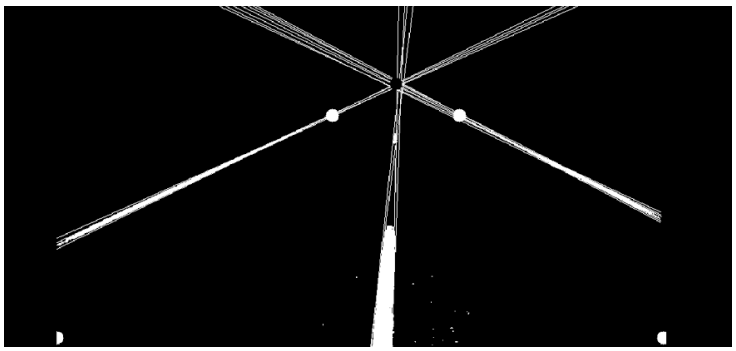


Ilustración 47. Resultado con punto de fuga.

En la imagen se puede observar un punto negro (que es el calculado a partir de la ecuación del punto de corte) y cuatro puntos blancos, situados a partir de este primero, necesarios para realizar la transformación a la vista de pájaro.

- **Transformación a vista de pájaro.**

Esta transformación significa simplemente un cambio de perspectiva de forma que podamos realizar un mejor análisis de la imagen tanto de las zonas más cercanas a nuestra visión como de las más alejadas.

El resultado será una imagen en la que observaremos las líneas paralelas entre sí como si las estuviésemos observando desde el aire.

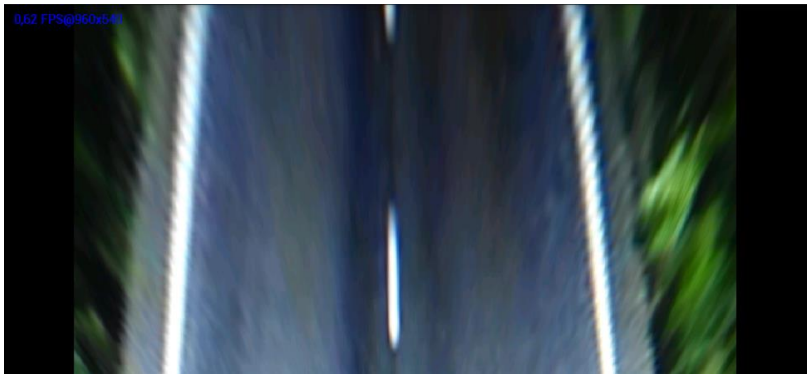


Ilustración 48. Resultado con vista de pájaro.

Como podemos apreciar en la imagen, en este momento se podrá realizar un análisis de ella con la misma fiabilidad tanto en las zonas cercanas a nuestra posición como de las más alejadas.

A pesar de no ser una imagen especialmente nítida, es más que suficiente para conseguir el objetivo marcado en un principio.

4.2.2 Resaltar y distinguir las líneas presentes en la imagen.

En esta segunda parte del código partimos de la imagen en vista de pájaro para llevar a cabo su análisis y conocer la continuidad y posición de las líneas de la carretera.

La estructura que se ha seguido es la siguiente:

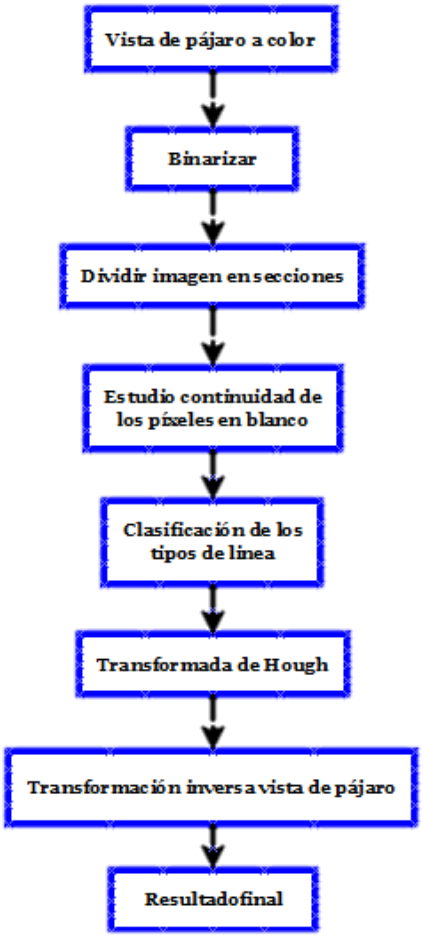


Ilustración 49. Estructura segunda parte del código.

- **Binarización de la imagen.**

Al igual que en la primera parte del trabajo, se binariza la imagen después de haber pasado previamente la imagen de color a escala de grises.

Ilustración 50. Vista de pájaro binarizada.

Como podemos ver, la información de la imagen es sencilla, intuitiva y la justa y necesaria para su correcto análisis. Llegado este punto en el que la información es lo más sencilla posible se puede pasar a su análisis con la seguridad de que el "ruido" presente en ella es el mínimo y no provocará errores.

- **Dividir imagen en secciones.**

Puesto que el punto de referencia para llegar a esta imagen es el hipotético punto de corte entre las rectas de la imagen original, hace que en nuestra imagen actual, cada una de las rectas ocupe siempre una posición parecida (con pequeñas variaciones) sea cual sea la imagen. Por ello se puede hacer un análisis por partes de ella.

Se dividirá la imagen en tres partes, asegurándonos de que en cada una de ella este contenida cada una de las líneas. Para un mejor resultado lo mejor es realizar pruebas con distintas imágenes para ajustarla de tal forma que solo haya una línea presente en cada una de las partes.

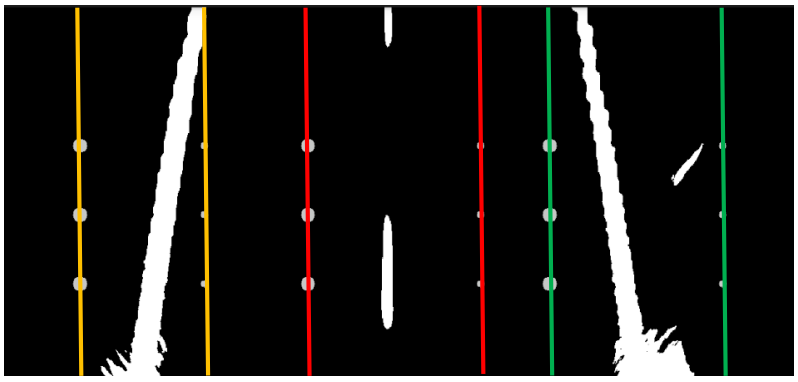


Ilustración 51. Imagen dividida en zonas para su posterior análisis.

- **Continuidad de los píxeles blancos y clasificación del tipo de línea.**

Una vez dividida la imagen se pasa a analizar cada parte por separado. Se trata de buscar píxeles blancos a distintas alturas y cada vez que encuentre uno, recibe un voto. Dependiendo del número final de votos que reciba, puede resultar ser una zona sin línea o tratarse de una línea discontinua o continua.

En base a ese análisis se mostrará un "semáforo" en la pantalla que indicará la condición de cada zona. Si se trata de una línea discontinua, la luz aparecerá en verde; de ser

continua aparecerá en rojo; y si no hay línea se omitirá la luz. Tal y como se muestra en la imagen:

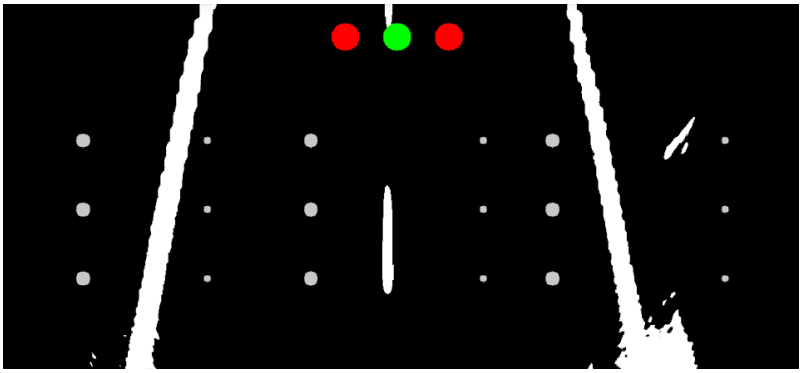


Ilustración 52. Clasificación del tipo de línea.

- **Transformada de Hough.**

En este caso la transformada de Hough, ya explicada anteriormente, servirá para resaltar toda la línea.

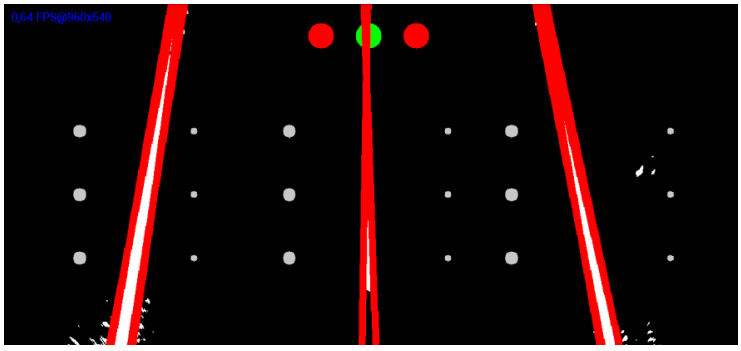


Ilustración 53. Clasificación de línea y transformada Hough.

No obstante, en algunos casos en los que la línea es discontinua la transformada de Hough no la detecta y no quedará resaltada. Algo que no supone demasiado problema ya que el análisis sobre si hay línea o no, se ha realizado en el paso anterior.

- **Transformación inversa de la vista de pájaro.**

Por último se deshará la transformación en vista de pájaro para obtener un resultado más cercano a la realidad en cuanto a perspectiva.

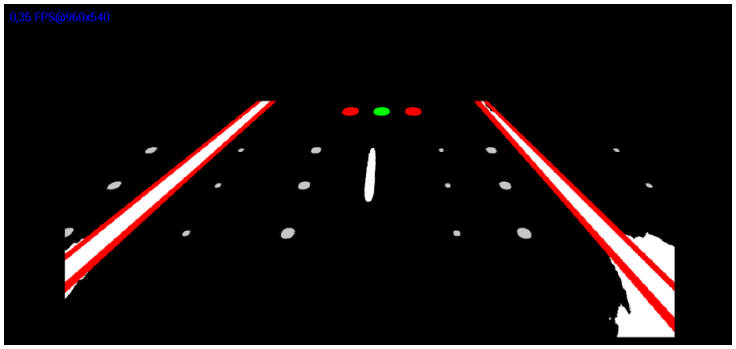


Ilustración 54. Resultado final.

Como podemos ver en este caso, la línea central no ha sido resaltada debido a lo anteriormente mencionado: su discontinuidad.

Llegados a este punto ya podemos conocer la posición exacta de las líneas presentes en la imagen y el tipo que son.

- **Añadido de los botones.**

Al tratarse de una aplicación en desarrollo y en prueba, en muchos casos el correcto funcionamiento depende de cómo se enfoque la carretera. Por ello se han añadido dos botones a la aplicación.

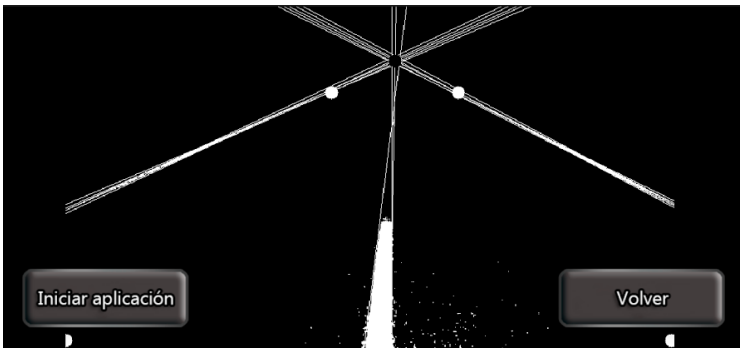


Ilustración 55. Aplicación con botones.

Al iniciar la aplicación observamos la imagen binarizada, con la transformada de Houhg y con el punto de fuga marcado. De esta forma, podemos enfocar la imagen hasta que consigamos el resultado deseado (buen posicionamiento del punto de fuga). Al tener una buena imagen, se presionaría sobre el botón "Iniciar aplicación" y nos mostraría la solución final explicada anteriormente (distinción entre tipo de líneas y resaltadas de rojo).

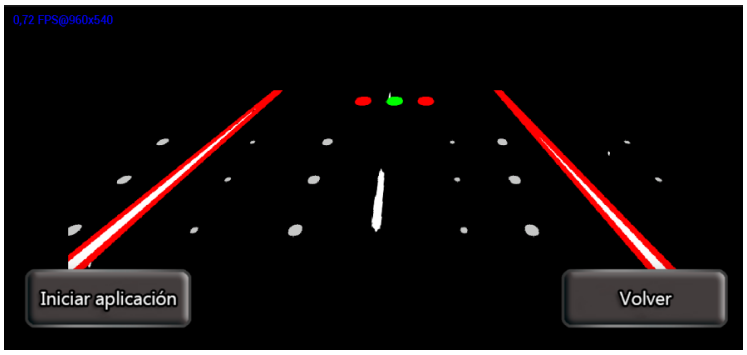


Ilustración 56. Resultado al presionar el botón "Iniciar aplicación".

En caso de que se desenfocara la imagen, se presionaría el botón "volver" y tendríamos la posibilidad de volver a buscar una correcta colocación del punto de fuga.

5. Evaluación, pruebas y limitaciones.

Para evaluar el funcionamiento de la aplicación, sus puntos fuertes, sus debilidades y posibles mejoras se han realizado pruebas con distintas imágenes de carreteras. Se ha hecho una selección de varías de ellas con alguna característica representativa de forma que se puedan sacar conclusiones sobre el funcionamiento de todas ellas.

Para la clasificación de cada una de las líneas se ha utilizado un pequeño "semáforo" en el que el color **rojo indica que la línea es continua** y el color **verde indica que la línea es discontinua**. Por otro lado en caso de que no aparezca en esos colores indicará que no detecta línea.

5.1 Prueba 1:

Ilustración 57. Prueba 1.

En esta imagen se puede apreciar un buen contraste entre todas sus partes (líneas, carretera, márgenes de la carretera, etc.). La distancia que hay entre sus líneas discontinuas es significativa por lo que no debería dar ningún problema a la hora de distinguirlas.

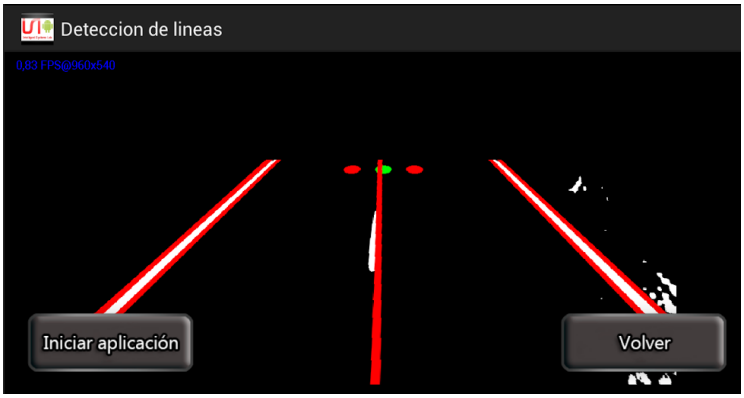


Ilustración 58. Resultado prueba 1.

Vemos que el resultado es correcto, detecta todas las líneas y las clasifica de forma correcta.

5.2 Prueba 2:

Ilustración 59. Prueba 2.

El resultado es el siguiente:

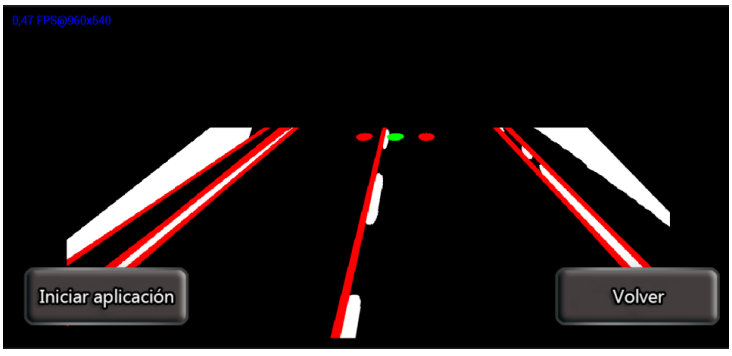


Ilustración 60. Resultado prueba 2.

Detecta y clasifica las líneas correctamente, no obstante, vemos como el bordillo izquierdo también es resaltado como línea cuando lo ideal sería que no lo hiciera. Sin embargo, al estar fuera de los límites de la carretera no generaría mayores problemas.

5.3 Prueba 3:

Ilustración 61. Prueba 3.

Cuyo resultado es:

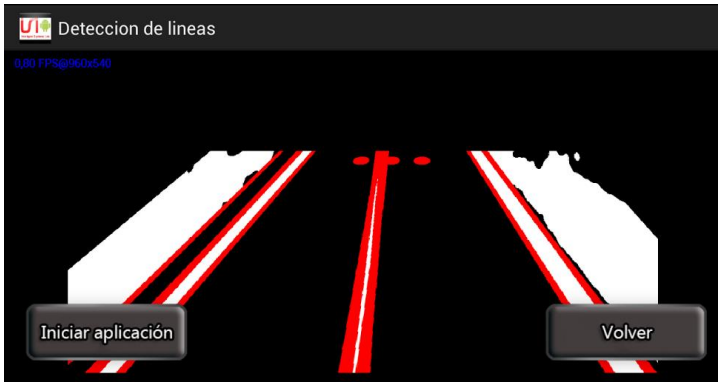


Ilustración 62. Resultado prueba 3.

Al igual que en las anteriores imágenes detecta bien las líneas, sin embargo, en este caso no hace una buena clasificación ya que las líneas discontinuas del medio presentan poco espacio entre ellas.

La forma de solucionarlo sería hacer un mayor muestreo en esa zona, es decir, en vez de analizar la línea a un número determinado de alturas, hacerlo a más, así será más probable que encuentre un hueco entre ellas y la clasificaría de forma correcta. Se trata de algo sencillo de implementar aumentar el número de alturas a la que analiza la línea.

5.4 Prueba 4:

Ilustración 63. Prueba 4.

En este caso, la línea del medio es continua, algo distinto a los casos anteriores:

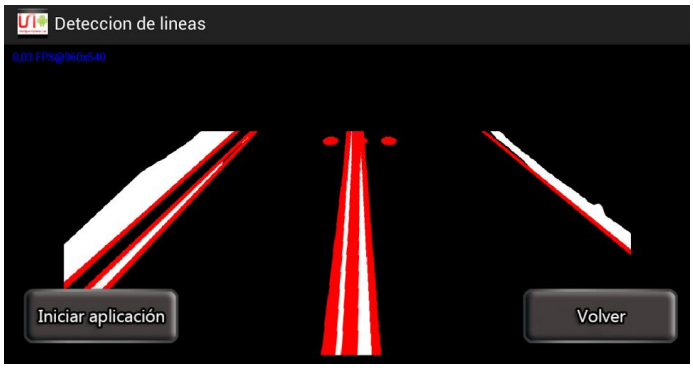


Ilustración 64. Resultado prueba 4.

La aplicación vuelve a reportarnos un resultado exitoso tanto a la hora de detectar líneas como a la hora de clasificarlas.

5.5 Pueba 5:

Ilustración 65. Prueba 5.

En esta imagen vemos como el asfalto presenta una rugosidad más marcada por lo que, en un principio, se puede pensar que genere demasiado ruido a la hora de detectar las líneas. Además no es tan oscuro como en los anteriores casos por lo que a la hora de binarizar, el valor límite para distinguir entre blanco y negro puede no ser el adecuado.

Observemos los resultados:

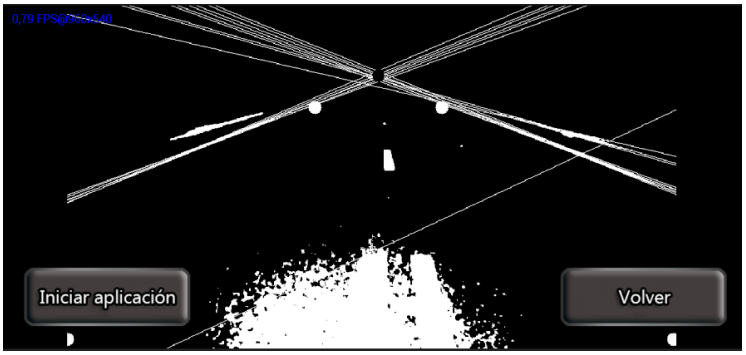


Ilustración 66. Resultado intermedio prueba 5.

A la hora de detectar las líneas vemos como el ruido generado por las diferentes texturas genera falsos positivos (línea donde no la hay), sin embargo, no logra crear un número significativo de falsos positivos como para provocar un mal funcionamiento de la aplicación en esta primera parte.

Veamos la segunda:

Ilustración 67. Resultado prueba 5.

El resultado no es el deseado. A la hora de binarizar, el valor que marca que píxeles deben ir en blanco y cuales en negro no es el adecuado para obtener un resultado óptimo ya que en esta imagen el color del asfalto es más claro que en los anteriores. Se trata de un error provocado por la iluminación.

Para subsanar este error se puede bajar el valor umbral de binarización (aunque sin excederse para no generar falsos positivos en imágenes con el asfalto más oscuro).

5.6 Prueba 6:

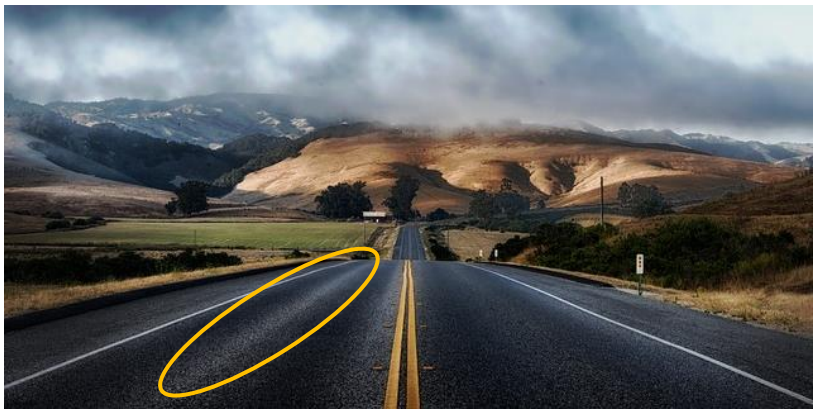


Ilustración 68. Prueba 6.

En esta imagen tenemos unas líneas bien señalizadas pero un asfalto irregular. La gran parte del asfalto es de un tono oscuro, sin embargo, en algunas partes (zona resaltada) el tono es más claro lo que podría generar un error.

Veamos el resultado:

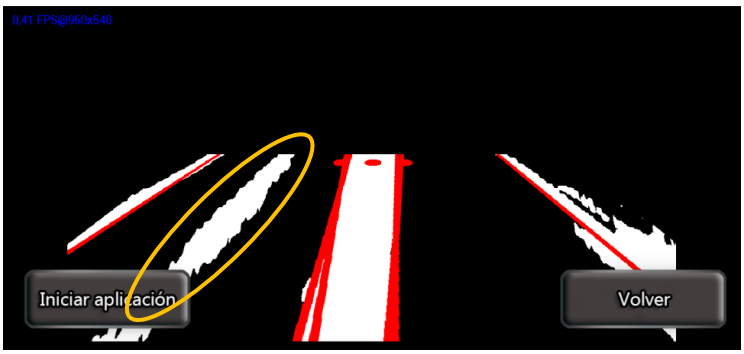


Ilustración 69. Resultado prueba 6.

La primera mitad del trabajo, la búsqueda de la vista de pájaro, la realiza correctamente, sin embargo, al realizar la umbralización y quedarnos únicamente con valores negros y blancos observamos un falso positivo. Este está generado por lo nombrado anteriormente, ciertas partes de la carretera tienen un tono más claro.

En este caso no afecta a su correcto funcionamiento ya que el resto de líneas son continuas, no obstante, en caso de que alguna de ellas fuese discontinua, tendríamos un mal resultado. El problema vuelve a ser relacionado con la iluminación y la solución seguiría siendo buscar otro valor umbral para binarizar la imagen.

5.7 Prueba 7:

Ilustración 70. Prueba 7.

La imagen es propicia a que se obtengan los mismos errores que en el anterior caso (carretera oscura y ciertas zonas más claras) pero agravadas por las condiciones de menor luz.

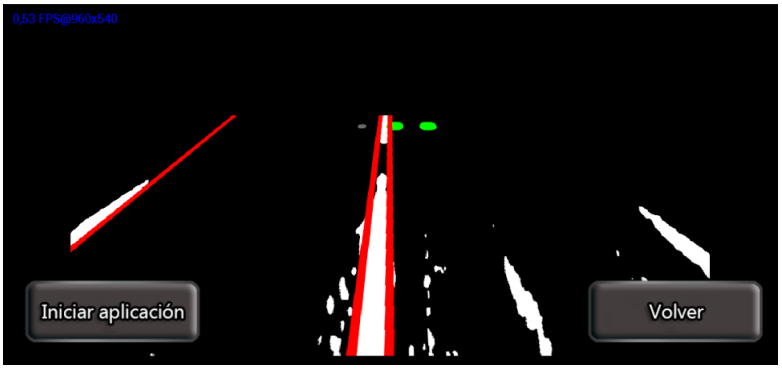


Ilustración 71. Resultado prueba 7.

Las zonas más claras sobre el asfalto no generan problemas, en este caso lo que provoca errores es la falta de iluminación de la imagen. Las líneas de los lados no son detectadas en su totalidad y por lo tanto no son clasificadas correctamente mientras que la del medio que resalta más si la clasifica de forma correcta.

5.8 Prueba 8:

Ilustración 72.Prueba 8.

Con esta imagen se han obtenido dos resultados dependiendo de la orientación de la cámara:

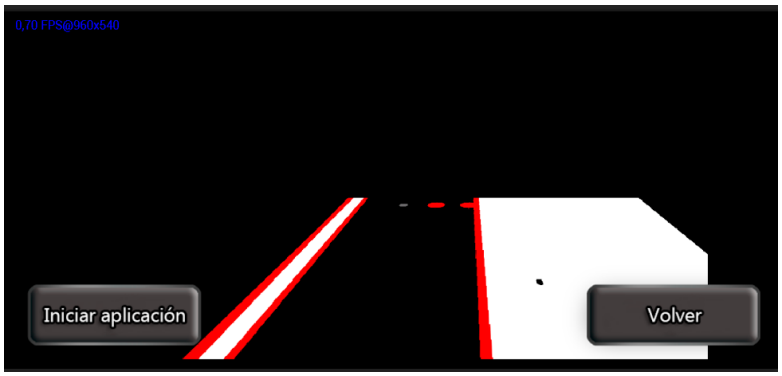


Ilustración 73. Resultado 1 prueba 8.

En este primer posible resultado la línea de la izquierda no es detectada ya que en la imagen original se distingue muy levemente del resto de la carretera. El resto de la información la recoge de forma correcta distinguiendo tanto la línea central como la de la derecha.

Ilustración 74. Resultado 2 prueba 8.

En este segundo caso se ha enfocado la cámara más hacia la izquierda de forma que la línea de ese lado también es detectada y distinguida correctamente. Por otro lado la línea izquierda sigue distinguiéndola de forma correcta mientras que la central si es detectada pero está mal clasificada. Esto es posiblemente debido a que, provocado por la inclinación, la línea central salga de la zona en la que es estudiada para su clasificación y en alguno de los puntos no detecte los píxeles blancos.

Para solucionar esto bastaría con reorganizar las zonas de análisis de cada línea. En este caso bastaría con aumentar la amplitud de la zona del medio.

5.9 Evaluación:

Si analizamos todos los resultados en global se pueden sacar distintas conclusiones:

5.9.1 Enfoque de la carretera:

En la mayoría de las imágenes observamos que están hechas desde un punto central de la carretera. Se puede pensar en un primer momento que no se trata de un caso real ya a la hora de circular se hará por uno de los carriles. En realidad la imagen no se aleja tanto de la realidad ya que ese problema se puede solucionar colocando la cámara en la parte del coche que mejor funcionamiento haga que tenga la aplicación, en este caso la izquierda.

Colocando la cámara en la parte izquierda del coche se tiene una imagen casi central de toda la carretera casi como el de la mayoría de las imágenes analizadas. No obstante, si la cámara se quisiese instalar en la parte central del coche bastaría con ajustar unos parámetros de la aplicación para que su funcionamiento fuese correcto.

5.9.2 Punto de fuga:

Para este apartado es importante que las líneas de la carretera estén bien visibles y tengan un buen contraste respecto del resto de la carretera. En todas las pruebas realizadas el cálculo del punto de fuga ha sido hallado correctamente por lo que los malos resultados obtenidos no son debido a esta parte.

En ciertas imágenes, calcula un número alto de líneas y en otras apenas encuentra una de cada lado. En ambos casos lo realiza de forma correcta ya que si encuentra varias únicamente trabaja con unas pocas y en caso de encontrar una de cada lado, trabaja solo con esas.

5.9.3 Iluminación:

Es el aspecto que más problemas crea en el correcto funcionamiento de la aplicación. Puede darse el caso de que se trabaje en unas condiciones óptimas de luz o en otras más desfavorables como las de una iluminación escasa o demasiado alta que pueda provocar reflejos. Son en estos dos últimos casos en los que más se puede mejorar la aplicación.

Como se ha comprobado en las distintas pruebas, las condiciones desfavorables de luz provocan que no se puede distinguir de forma correcta entre los distintos tipos de líneas.

6. Trabajos futuros.

6.1 Mejora de los resultados.

Como primer posible futuro trabajo se podría realizar una mejora de los resultados obtenidos, especialmente los relacionados con la iluminación inadecuada.

Se trabajaría en conseguir adquirir una imagen con características similares (de luminosidad y contraste) sea cuales sean las condiciones de luminosidad. A través de la cámara se podría conocer si hay demasiada o insuficiente luz y a raíz de ese resultado realizar las adaptaciones necesarias para evitar variaciones de resultados según la luz.

6.2 Mayor rapidez de procesado.

Otro de los puntos en los que se puede mejorar es este apartado. Ya sea mediante mejoras de hardware o de software es necesario una mayor rapidez de trabajo especialmente por el área al que va encaminada la aplicación, la seguridad vial. Cuanto más rápido trabaje mayor será la seguridad que genere el programa.

6.3 Adaptación a más carriles.

Se ha trabajado en todo momento con carreteras de dos carriles por lo que se trata de otro punto a mejorar. La imagen, para distinguir entre los tipos de líneas, se ha separado en tres partes de forma predeterminada y según lo que se encuentre en ellas clasifica las líneas.

Una solución podría ser que, automáticamente, distinguiera el número de grupos de líneas presentes en la imagen y a partir de ellas crear tantas partes en la imagen como sean necesarias.

6.4 Unificación con otras aplicaciones del mismo campo.

La aplicación desarrollada mejorará la seguridad vial en cierto nivel, sin embargo, en la carretera hay más peligros a parte de los relacionados con este trabajo. El siguiente paso podría ser unificar las distintas aplicaciones relacionadas con el área (detección de señales de tráfico, proximidad a otros coches, etc.) para conseguir una seguridad mayor y más completa.

6.5 Comunicación aplicación-vehículo.

Conseguir comunicar la aplicación con los sistemas electrónicos del vehículo podría conseguir que el programa pudiera adelantarse a nuestros movimientos y aumentar el tiempo de reacción que el conductor pueda tener.

Por ejemplo, en caso de activar un intermitente del vehículo, que esa señal fuera comunicada a nuestra aplicación y en caso de estar detectando una línea continua hacia ese lado, nos avisara de la infracción que se está a punto de cometer.

7. Conclusiones.

7.1 Viabilidad en Android.

Como se ha demostrado a la largo del trabajo, el desarrollo de aplicaciones para la seguridad vial es plausible para Android y dispositivos tan comunes hoy en día como los SmartPhones.

Aunque en la actualidad se están desarrollando cámaras y dispositivos específicamente para llevar a cabo esta labor, la opción de poder tener las mismas prestaciones en nuestro dispositivo Android hace que pueda estar al alcance de un mayor número de personas. Se trata de una opción mucho más económica.

En un principio pueden haber más limitaciones trabajando con estos dispositivos pero, con un mayor desarrollo del hardware y simplificando al máximo el software se podrían conseguir resultados más óptimos que hagan de esta una muy buena opción.

7.2 Dificultad por el gran número de variables presentes en la carretera.

En condiciones ideales (correcta iluminación, buen contraste, ausencia de otros coches en la carretera, etc.) el desarrollo de este tipo de aplicaciones sería mucho más sencillo. Pero la realidad es bien distinta, lo normal es que las condiciones ideales se den muy pocas veces, por eso, es importarte seguir trabajando en este aspecto y evitar que este tipo de factores no impidan el buen funcionamiento de este y otro tipo de aplicaciones del mismo área.

Quizá este sea uno de los impedimentos más grandes en este campo, controlar todas las situaciones que puedan darse y actuar en consecuencia. Algo de vital importancia ya que en juego está la seguridad de las personas.

7.3 Utilidad de las aplicaciones para la seguridad vial.

Con lo desarrollado a lo largo de este trabajo y el resto de ejemplos vistos y explicados se puede sacar en conclusión que la combinación de todas ellas pueden evitar un gran número de accidentes.

La aplicación de cada una de ellas por separado puede resultar útil pero todas ellas juntas resultan de mayor utilidad. No solo con el objetivo de minimizar el número de accidentes sino con el de conseguir vehículos autónomos que, perfeccionándolos, puedan eliminar por completo los accidentes, pero hasta que eso ocurra es importante desarrollar todas las ayudas posibles a la conducción.

7.4 Futuro de la visión por computador aplicada a la seguridad vial.

El número de aplicaciones de la visión por computador están en auge por la cantidad de posibilidades que puede darnos aunque a la vez presenta una gran dificultad, entre otras cosas, por lo ya comentado de las numerosas variables y la infinidad de situaciones que nos podemos encontrar.

Cada vez salen al mercado más vehículos que incluyen estas tecnologías lo que hace pensar que tendrán una gran aceptación e impulsará un mayor desarrollo.

7.5 Extensión de las aplicaciones para Android en otros campos.

Que sea posible la aplicación de esta tecnología en algo tan complicado como el tema de la seguridad vial, hace pensar que se podría aplicar con mayor facilidad en otros ámbitos como, por ejemplo, la agricultura en maquinaria guiada de forma autónoma gracias a la información captada a través de una cámara.

También la posibilidad de desarrollo de infinidad de aplicaciones a nivel usuario gracias al fácil acceso de cualquier persona a un SmartPhone Android lo que permitiría un mayor intercambio de información y, por lo tanto, un desarrollo de la visión artificial más rápido.

8. Presupuesto.

Para calcular el presupuesto total del proyecto se tendrán en cuenta por un lado los materiales y por el otro el personal humano.

8.1 Presupuesto de materiales.

Para el cálculo de este apartado no es suficiente con conocer el precio total de los materiales, es necesario conocer la amortización de cada uno de ellos.

Según la nueva Ley aprobada el 27 de Noviembre de 2014 con fecha de entrada en vigor el 1 de Enero de 2015 del Impuesto sobre Sociedades la amortización para equipos electrónicos destinados a procesos de información es de un 25% en un total de 8 años [25].

Para calcular el coste total se hará siguiendo la siguiente fórmula:
Coste de amortización = (Precio del bien*coeficiente de amortización*meses)/12
donde el coeficiente de amortización será del 25% y los meses serán 7.

Coste total de materiales				
Material	**Descripción**	**Precio total**	**Coste de amortización**	**Coste en el proyecto**
Ordenador	Toshiba corei5 con Windows	559,21	(559.21*0,25*7/12	81,55
SmartPhone	BQ Aquaris 5 HD con Android	207	(207*0,25*7)/12	30,19
			TOTAL	111,74

Tabla 1. Presupuesto materiales.

8.2 Presupuesto de personal.

Para la elaboración de este apartado se asume que la persona encargada del trabajo tiene un contrato en prácticas, lo que significa que como mínimo debe cobrar un 60%

del sueldo medio fijado en el convenio del trabajador que realizaría dicho trabajo. En este caso el sueldo es de 22.790,10€ [26].

Por lo tanto para el cálculo del salario se seguirá la siguiente expresión:

Coste salario personal = (sueldo base*7*0.6)/12

Esto asumiendo que el sueldo recibido por la persona en prácticas es el mínimo posible.

Coste total de personal			
Sueldo bruto anual	**Duración**	**Contrato en prácticas**	**Sueldo total**
22.790,10€	7 meses	60%	7.976,54

Tabla 2. Presupuesto de personal.

8.3 Presupuesto total.

El coste total del proyecto queda resumido en la siguiente tabla:

Bloque	*Coste*
Materiales	111,74
Personas	7.976,54
Total	8.088,28

Tabla 3. Presupuesto total.

9. Normativa.

Las licencias utilizadas necesarias para el desarrollo del proyecto son las referentes a los programas OpenCV, Eclipse y Android:

- Licencia BSD para OpenCV [27]. Se trata de una licencia que permite que sea usado libremente tanto para propósitos comerciales como para propósitos de investigación bajo las condiciones que en ella se expresan.
- Eclipse Public License [28]. Es una licencia de software de código abierto utilizada por la fundación Eclipse para su software y sustituye a la Licencia Pública Común (CPL). Permite utilizar, modificar, copiar y distribuir el trabajo y las versiones modificadas.
- Licencia Apache 2.0 para Android [29]. Es una licencia de software libre creada por la Apache Software Foundation (ASF). Permite al usuario del software la libertad de usarlo para cualquier propósito, distribuirlo, modificarlo y la distribución de versiones modificadas.

10. Bibliografía.

[1] «OpenCV. Open Source Computer Vision,» [En línea]. Available: http://opencv.org/. [Último acceso: Julio 2015].

[2] «Aula Clic,» [En línea]. Available: http://www.aulaclic.es/articulos/android.html. [Último acceso: Julio 2015].

[3] R. M. Prieto Galarza. [En línea]. Available: http://www.monografias.com/trabajos97/inteligencia-artificial-algoritmos-geneticos/inteligencia-artificial-algoritmos-geneticos.shtml. [Último acceso: Julio 2015].

[4] «APLICACIONES DE LA VISIÓN ARTIFICIAL,» [En línea]. Available: http://dmi.uib.es/~ygonzalez/VI/Material_del_Curso/Teoria/Aplicaciones_VC.PDF. [Último acceso: Julio 2015].

[5] E. de la Fuente, «Librovision,» [En línea]. Available: http://www.librovision.eii.uva.es/pdf/cap10.pdf. [Último acceso: Julio 2015].

[6] «Motorpasion,» [En línea]. Available: http://www.motorpasion.com/seguridad/volvo-esta-investigando-sistemas-de-reconocimiento-facial-para-evitar-las-distracciones-del-conductor. [Último acceso: Julio 2015].

[7] J. Javier Yebes, L. M. Bergasa y P. F. Alcantarilla, «Robesafe,» [En línea]. Available: http://www.robesafe.com/personal/javier.yebes/docs/Yebes09WAF.pdf. [Último acceso: Julio 2015].

[8] «Circulo seguro,» [En línea]. Available: http://www.circulaseguro.com/que-es-el-ldw-o-detector-de-cambio-de-carril/. [Último acceso: Julio 2015].

[9] RACC - ADAC, «SISTEMAS DE RECONOCIMIENTO DE SEÑALES DE TRÁFICO EN TURISMOS,» [En línea]. Available: http://imagenes.w3.racc.es/uploads/file/22207_Sistema_Reconocimiento_Senyales.pdf. [Último acceso: Julio 2015].

[10] «Motor pasión futuro,» [En línea]. Available: http://www.motorpasionfuturo.com/ayudas-a-la-conduccion/continental-contiguard-prevencion-de-accidentes-mediante-la-deteccion-de-objetos. [Último acceso: Julio 2015].

[11] «km77,» [En línea]. Available: http://www.km77.com/00/bmw/5/2010/t05.asp. [Último acceso: Julio 2015].

[12] A. Campos Albuixech, «Universidad Politécnica de Valencia,» [En línea]. Available: https://riunet.upv.es/bitstream/handle/10251/46232/Memoria.pdf?sequence=1. [Último acceso: Julio 2015].

[13] «Motor pasión futuro,» [En línea]. Available: http://www.motorpasionfuturo.com/coches-del-futuro/como-funciona-el-coche-autonomo-de-google. [Último acceso: Julio 2015].

[14] «Motor pasión futuro,» [En línea]. Available: http://www.motorpasionfuturo.com/ayudas-a-la-conduccion/las-nuevas-luces-de-opel-iluminaran-alla-donde-miremos. [Último acceso: Julio 2015].

[15] «iOnRoad,» [En línea]. Available: http://www.ionroad.com/spa. [Último acceso: Julio 2015].

[16] V. Cancho Armesto, «BIBLIOTECA PARA LOCALIZACIÓN DE,» Universidad Carlos III, Madrid.

[17] J. Capra, B. Capra, C. Aciti y J. Marone, «Instituto INTIA, Argentina,» Octubre 2012. [En línea]. Available: http://sedici.unlp.edu.ar/bitstream/handle/10915/23802/Documento_completo.pdf?sequence=1. [Último acceso: Septiembre 2015].

[18] L. M. Gracia. [En línea]. Available: https://unpocodejava.wordpress.com/2013/10/09/que-es-opencv/. [Último acceso: Julio 2015].

[19] Á. Carbajo Benito, «Desarrollo de una aplicación Android para la detección de señales de tráfico,» Universidad Carlos III, Madrid, 2015.

[20] D. Arroyo Cazorla, «Adquisición de datos remota mediante el sistema OBD-II y dispositivo móvil,» Universidad Carlos III, Madrid, 2014.

[21] A. Ramos López, «Implementación de Algoritmos de Visión por Computador en Plataforma Android,» Universidad Carlos III, Madrid, 2014.

[22] I. developerWorks, «IBM,» [En línea]. Available: http://www.ibm.com/developerworks/ssa/library/os-ecov/. [Último acceso: Julio 2015].

[23] J. Valverde Rebaza, «Detección de bordes mediante el algoritmo de Canny,» Universidad Nacional de Trujillo .

[24] J. F. López, «Procesamiento digital de imágenes,» [En línea]. Available: https://procesamientodigitalimagenes.wordpress.com/2012/11/02/transformada-hough/. [Último acceso: Julio 2015].

[25] «GABILOS, SOFTWARE DE GESTIÓN PARA PYMES Y PROFESIONALES,» [En línea]. Available:

http://www.gabilos.com/webcontable/amortizacion/estimacion_directa_simplificada.htm. [Último acceso: Agosto 2015].

[26] «CONETIC,» [En línea]. Available: http://www.aeiciberseguridad.es/descargas/categoria6/8774932.pdf. [Último acceso: Agosto 2015].

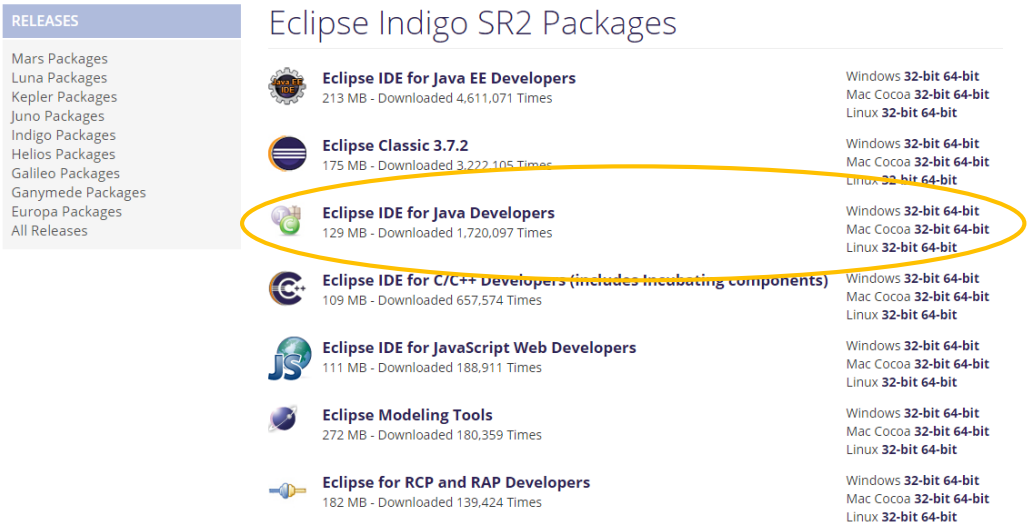
[27] G. Lehey, «freebsd.org,» [En línea]. Available: https://www.freebsd.org/doc/es_ES.ISO8859-1/articles/explaining-bsd/article.html. [Último acceso: Septiembre 2015].

[28] «Eclipse.org,» [En línea]. Available: https://www.eclipse.org/legal/epl-v10.html. [Último acceso: Septiembre 2015].

[29] «source.android.com,» Android, [En línea]. Available: https://source.android.com/source/licenses.html. [Último acceso: Septiembre 2015].

ANEXO I. Instalación y configuración del entorno de trabajo.

En primer lugar, se instalará Eclipse desde el siguiente enlace: http://www.eclipse.org/downloads/packages/release/Indigo/SR2

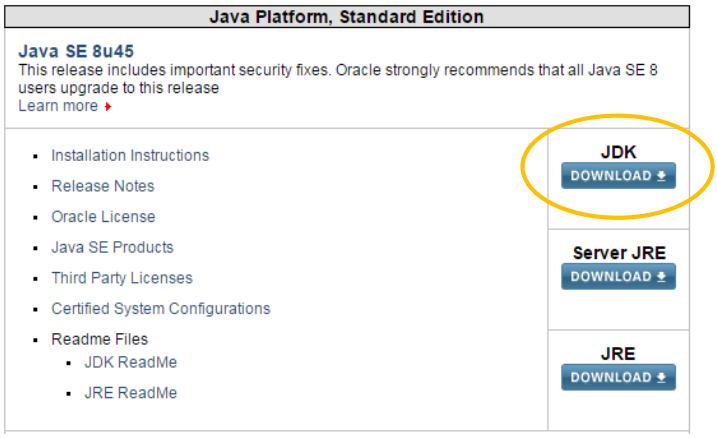


Nos quedaremos con la versión para Java Developers e instalaremos la versión compatible para el ordenador desde el que trabajaremos.

JAVA DEVELOPMENT KIT (JDK)

Java Development Kit o JDK son un conjunto de herramientas (programas y librerías) que permiten desarrollar (compilar, ejecutar, generar documentación, etc.) programas en lenguaje Java.

Será descargado desde el siguiente enlace: http://www.oracle.com/technetwork/java/javase/downloads/index.html



Y a continuación seleccionar la versión compatible con el ordenador desde el que se trabajará.

Software Development Kit (SDK)

Se trata de un paquete de software liberado por Google que nos ayudará a realizar aplicaciones y programas para la plataforma Android o incluso instalar un emulador para correr de manera virtual el sistema operativo creado por la compañía del buscador específico para dispositivos móviles.

Para descargarlo vamos al enlace siguiente: http://developer.android.com/sdk/index.html y escogemos la opción compatible a nuestro sistema operativo

Platform	Package	Size	SHA-1 Checksum
Windows	installer_r24.3.3-windows.exe (Recommended)	139463749 bytes	bbdae40a7665e55b3cdb1fbae865986e6cd3df14
	android-sdk_r24.3.3-windows.zip	187480692 bytes	b6a4899efbf20fc593042f1515446c6630ba502e
Mac OS X	android-sdk_r24.3.3-macosx.zip	98330824 bytes	41f0f3e76d6868018740e654aefb04fd765c357d
Linux	android-sdk_r24.3.3-linux.tgz	309109716 bytes	cd4cab76c2e3d926b3495c26ec56c831ba77d0d0

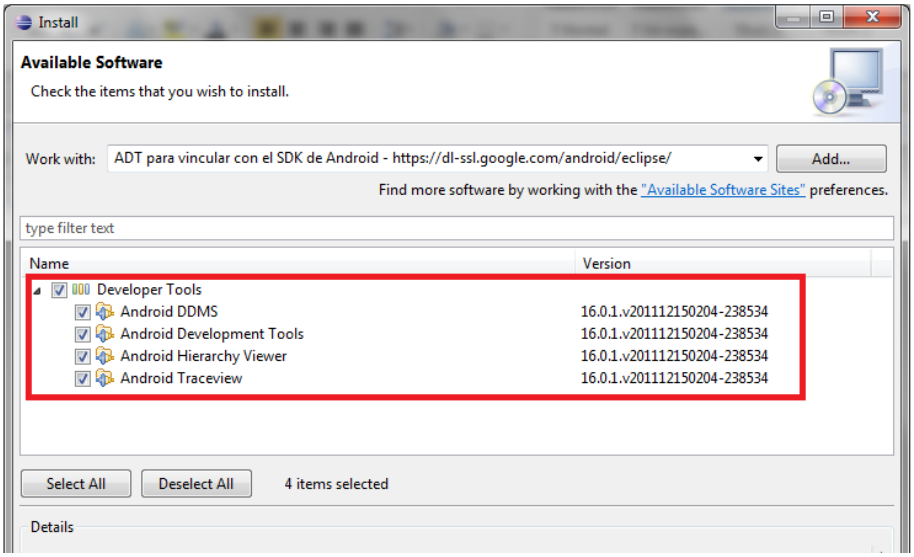
ADT (Android Developent Tools)

Es un plugin para Eclipse necesario para vincular el SDK con Eclipse y así poder trabajar con Android.

Desde Eclipse y siguiendo la rutina: "Help" – "Instaling new software" – "Add" le damos el nombre que queramos y en el campo URL se pone la siguiente:

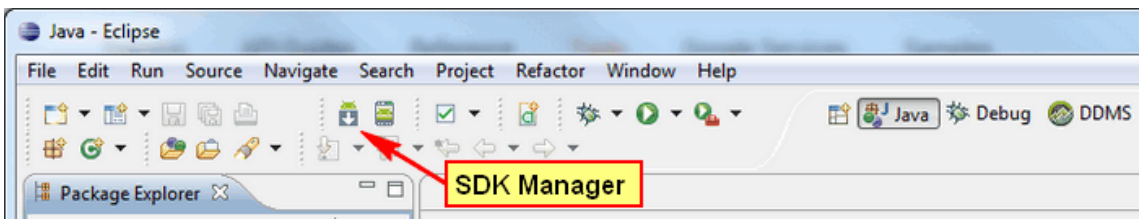
- https://dl-ssl.google.com/android/eclipse/

Y se marca la casilla "Developer Tools":

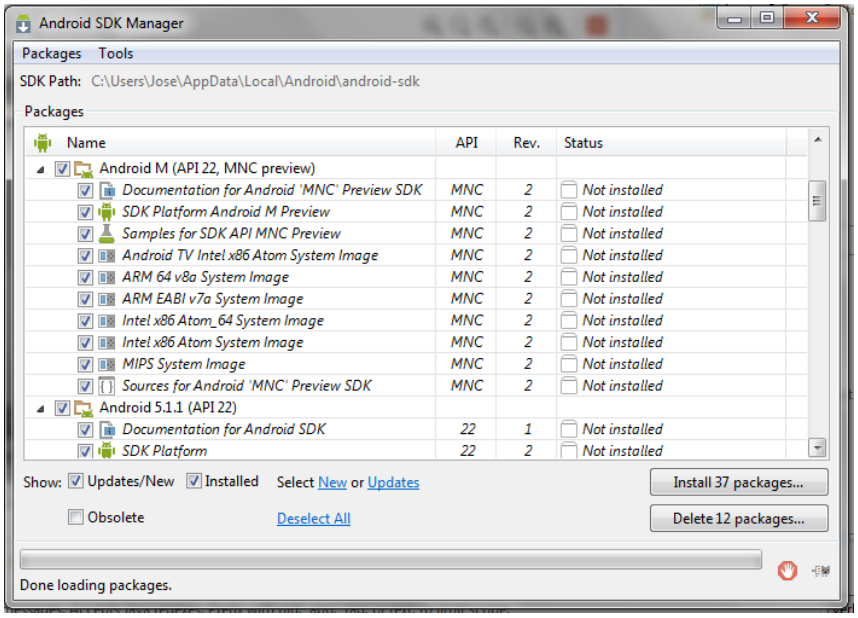


Se aceptan el resto de pasos y de esta forma queda vinculado el SDK de Android con Eclipse.

Por último, desde Eclipse y ya con el SDK vinculado se deberán descargar las APIs (o versiones de Android)



En él aparecerá un listado con todas las versiones del sistema operativo Android y, en función de la versión que se quiera desarrollar se deberá marcar una u otra.



Variables de entorno

Para configurar la variable de entorno *JAVA_HOME* se realiza lo siguiente:

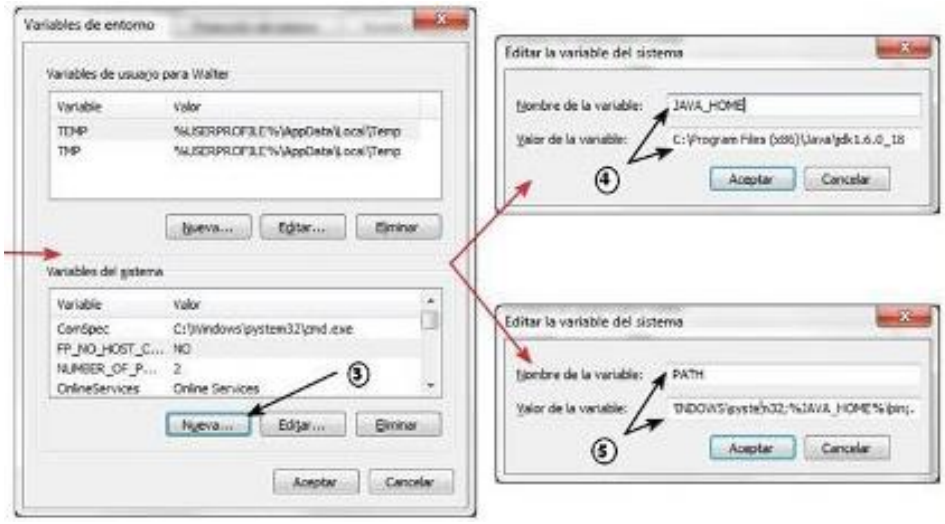
Abrimos el explorador de Windows o pulsamos sobre “Mi Pc”. Pulsamos sobre Equipo, Propiedades y con botón derecho del ratón o buscando el icono -> Configuración avanzada / Cambiar configuración -> Opciones avanzadas -> Variables de entorno -> Nueva… (Variables del sistema) e introducimos lo siguiente:

- Nombre de la variable: JAVA_HOME
- Valor de la variable: Ruta de donde está instalado el JDK de java, por defecto está en C:Program FilesJavajdk

También se deberá configurar la variable PATH que es donde se encuentran los ejecutables de Java necesarios para su ejecución. Siguiendo la misma rutina anterior se introduce lo siguiene:

- Nombre de variable: PATH

- Valor de variable: C:\WINDOWS;C:\WINDOWS\system32;%JAVA_HOME%\bin

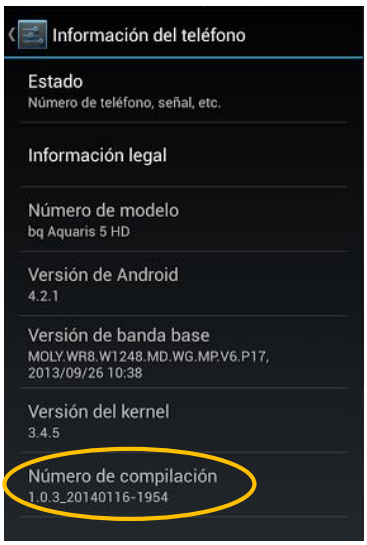


Depuración USB

El modo Depuración USB (USB Debugging) del dispositivo móvil Android está pensado principalmente para desarrolladores. El modo Depuración USB abre el acceso directo al sistema para el SDK de Android (Software Development Kit). Si descargamos e instalamos en nuestro ordenador el SDK de Android pero no activamos el modo depuración el SDK no funcionará, ya que no podrá conectar con el dispositivo.

Para habilitar la depuración de nuestra versión de Android (Jelly Bean y posteriores) habrá que ir a **Ajustes** de nuestro dispositivo móvil y a **Información del teléfono**.

A continuación pulsaremos varias veces sobre la opción **Número de compilación**.



Y por último en la pantalla anterior seleccionamos **Opciones de desarrollo** y activamos la opción de **Depuración USB**.